Novel Algorithms for Tracking Small and Fast Objects in Low Quality Images

---

A thesis
submitted in partial fulfilment
of the requirements for the Degree
of
Master of Science in Computer Science
in the
University of Canterbury
by
Hongzhi GAO

Examining Committee

Richard GREEN	Supervisor
Mark BILLINGHURST	Supervisor

University of Canterbury
2005

Abstract

In conventional computer vision systems, high image quality and long target exposure requirements are required. In this thesis, two algorithms to overcome such limitations of current computer vision systems have been proposed. The Pixel Exclusion Double Difference Algorithm (PEDDA) algorithm is a novel object detection algorithm that is able to detect fast moving objects in noisy images and suppress interference from large, low speed moving objects. The State-based "Observation, Analysis and Prediction" Target Election and Tracking Algorithm (SOAPtet) algorithm uses a deterministic state machine to guide the SOAPtet algorithm predictions. A novel stochastic based approach is also implemented in this algorithm to elect the target of interest from its candidates that are usually triggered by noise. A real time experimental system is developed based on the two algorithms. The experiment results show that this system detects up to 92.3% of moving objects in noisy environment and the tracking accuracy is up to 97.42%.

Table of Contents

Chapter 1

Introduction

Computer vision based target tracking algorithms have been intensively researched in recent literature, such as [7] [34 - 39] [43] [44] [46 - 48] [65 - 68] <perhaps just put the top five or so refs here and always list each number separately e.g. [7,34,35,36,37,38,39]>. Many computer vision based applications are developed based on established tracking algorithms. These applications can be divided into four categories based on their design criteria. They are: video surveillance applications, sport training applications, human-computer interaction applications and entertainment applications.

Computer Vision based Systems

Video surveillance applications usually utilize one or more stationary cameras to monitor a particular area, such as subway stations [65] or roads [43]. Normally, the tracking targets of these systems are people or vehicles. Such systems record the instant trajectory of each target [65]. Some systems in this category also contain an alarm subsystem. If an abnormal behaviour is detected, the system will alert alarm information to the human operator. The difficulties of current video surveillance systems are:

- Target occlusion

- Changing illumination
- Poor visibility (bad weather in outdoor systems)

Sport training applications use computer vision techniques to detect and track the trajectories of players [63] or balls [24] [35] [36] [39]. The trajectories are recorded and often manually used by a coach to support her or his training. There are two common sources of images for these applications. Some applications use a broadcast television signal as the source of image sequences. Others use stationary cameras mounted in the training environment to capture image sequences. The difficulties of these systems are:

- Target occlusion (when tracking athletes)
- High velocity targets (especially tracking balls)
- Indistinguishable targets (sometime the balls have similar colour to the background)
- Image registration (if the video source is broadcast television signal, the pan or zoom operations have to be compensated before doing any computer vision processing).

Computer vision based tracking techniques are used in human-computer interaction applications to detect and track natural features of a human body [67]. For example, human face or fingertips may be targeted by these systems to implement some body position awareness interface design. Normally, low cost web cameras are used as video source of the systems belonging to this category. The difficulties of this class of systems are:

- Low cost cameras have low signal-to-noise ratios.
- These systems are usually working in a cluttered background environment.
- The human operators usually expect a higher precision than the system can achieve.

Computer vision based systems are increasingly used in entertainment applications as well [68]. The applications belong to this category may range from cheap personal computer games to expensive human tracking systems in the movie industry [48]. Video input devices used in such systems ranges from low cost web cameras to high-end professional video devices. The challenges of such systems differ from project to project. Usually, in a low cost system, the challenge is to satisfy users within a limited budget and for a high-end system, to obtain near perfect tracking results without imposing too many limitations on the human performers and performance space.

Project Goal

Although the computer vision systems are well studied, there are many areas of active research. Firstly, computer vision based systems seldom utilize wireless cameras as their image source because the low quality of images captured by a wireless camera. However, a wireless camera has its own benefits. For example, a wireless camera is easy to deploy to remote locations, which would otherwise be difficult to access by a wired camera. Secondly, most of the computer vision based tracking algorithms require that the targets can be observed by the camera for a long duration. For example, in a sport training system, a ball has to be in the field of view in most of the

successive frames being analysed. Even in vehicle tracking systems, there is usually a brief window of time that the vehicles can be observed by the camera.

In this thesis, we propose a set of computer vision algorithms that can be used to track small and fast targets in noisy images. The appearance of the tracking targets in video may be as brief as only a few frames. The tracking algorithms proposed in this thesis will successfully track targets that appear within a couple of frames in a noisy environment.

Structure of the Thesis

This thesis consists of 7 chapters. In chapter 2, related research is discussed and relevant background knowledge is introduced. Chapter 3 gives an overview of this project including the problems, environment and solution approach. In chapter 4, two novel object detection algorithms developed for this project are discussed. In chapter 5, a novel target election and tracking algorithm is discussed. Chapter 6 discusses an experimental system based on the selected algorithms. Detection experiment, tacking accuracy experiment and system efficiency experiment are also introduced in this chapter. Finally, chapter 7 concludes the thesis and discusses the potential for future work.

Chapter 2

Literature Review

2.1 Introduction

In this chapter, several computer vision based object tracking systems are reviewed. The goals of these projects as well as their implementation methods and major limitations are discussed. Furthermore, as certain object detection techniques and target tracking techniques are closely related to this research, they are also presented in detail.

2.2 Tracking Systems Survey

In this section, recent computer vision based object tracking research will be discussed. These projects are classified into several types: video surveillance systems, sport training systems, human-computer interaction systems and entertainment systems.

Video Surveillance Systems

Computer vision techniques are used in video surveillance systems to automatically detect and track targets of interest, such as people. Here, two typical video surveillance systems are introduced.

The "Annotated Digital Video for Intelligent Surveillance and Optimised Retrieval" (ADVISOR) system (Figure 2-1) is a video surveillance system developed by Siebel and Maybank [65]. The goal of this project is to detect and track people in video sequences taken from indoor environments.

Figure 2-1: The ADVISOR computer vision video surveillance system [65].

The motion detector subsystem implemented in the ADVISOR system detects people by using a background subtraction technique. In this detector, the background is modelled by the average intensity values of a sequence of images that contain the appearance of the scene without people. The detector subtracts the background from the current video image. Then the difference images are thresholded to yield binary masks that contain “foreground” regions.

The region tracker subsystem tracks all moving regions detected by the motion detector using a frame-to-frame region matching algorithm. This algorithm matches regions by their similarities in size and position. Long-term trajectory histories are also taken into account when making a matching decision to solve occlusion problems. In addition, a delayed response technique is designed in the region tracker subsystem to solve the lost tracking problem. This technique is designed such that even if a track is not detected in one frame, or the video image is unavailable for a frame or two, their data will not be removed from the Tracking Status and History database (TSHD). Therefore, each tracker still makes predictions for the object in question, and the tracking may be re-gained at a later stage.

However, this system has the following limitations:

- The ADVISOR system has to work in a constrained environment with well controlled illumination.

- The size and position based region matching strategy works well for large and slow moving objects only.
- The system works at 5 frames per second (fps) which is not in real-time (usually 25 fps).

Malley *et. al.* [66] present a computer vision based video surveillance system (Figure 2-2), which tracks and identifies moving persons from video images taken by a fixed field-of-view camera. In this system, the moving person is tracked, and then their clothes colours are represented by a mixture of Gaussian models and preserved in a database for future identification.

Figure 2-2: The computer vision based video surveillance system developed by Malley *et. al.* [66].

To detect the people in the scene, this system models each pixel of a sequence of pure background images that contain only static features into a Gaussian distribution in HSV colour space. Foreground objects are detected by thresholding the Mahalanobis distance between a pixel's colour in the current frame and its background model.

In the tracking subsystem of this system, an initial state is implemented to distinguish targets of interest from noise. This initial state is known as the Track Initiation Processor (TIP) in this system. TIP classifies target of interest and noise based on the following three assumptions:

- Noise is randomly distributed through the scene.
- A person moving through the scene moves with linear velocity over short distances.
- A person moving through the scene moves with a limited maximum velocity.

While the target of interested is selected by TIP, a Kalman filter is utilized <use only NZ spelling in this NZ thesis – some examiners are fussy about this> to predict the position of this target over frames.

The limitation of this research is:

- A period of dedicated background modelling time is required before the system can begin to operate. No foreground moving objects are allowed to be observed in this period. This requirement is difficult to fulfil especially when the system is to be deployed in unconstrained public places.

Sport Training Systems

The following pages discuss recent research into computer vision techniques used in computer-aided sport training applications. In such systems ([34] [35] [36] [37] [38] [39] [47]), the trajectory information of a ball or an athlete is generated by computer vision based tracking algorithms and this information can be used to evaluate and improve the performance of the athlete.

Kovacic and Pers [47] have developed a computer vision based handball tracking application. In this system, two stationary cameras are mounted directly above the court (Figure 2-3) to make sure all players can be observed by the cameras from the beginning to the end of the game.

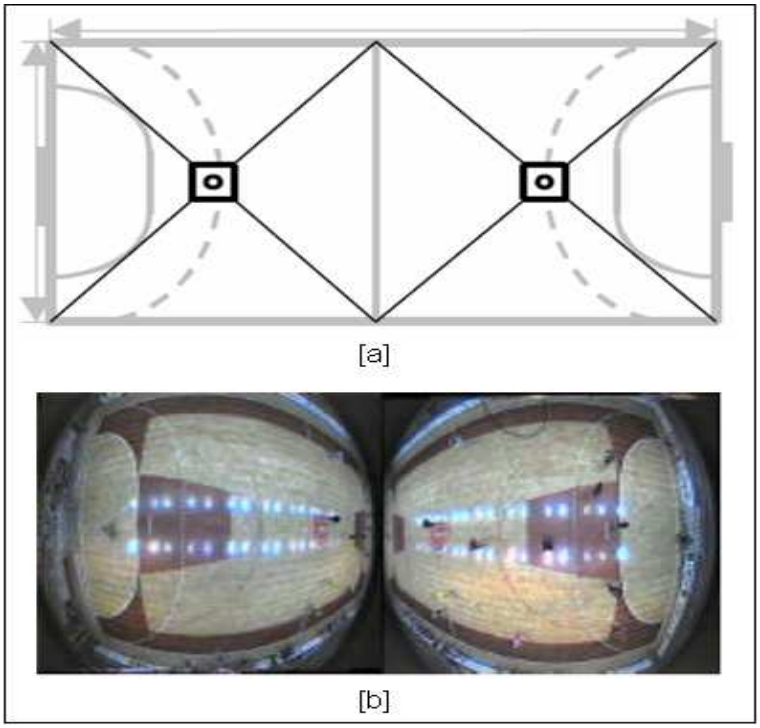


Figure 2-3: The handball tracking system. [a] Two cameras are mounted directly above the court. [b] An integrated image captured by both cameras [47].

The background is modelled using the first frame of a video sequence since this frame usually contains only common static background information. The players are detected by subtracting the background from the current frame. In addition, player tracking is done by temporal and colour matching processes performed within an image area surrounding the target's position in the last frame. The trajectory of each player can be plotted (Figure 2-4) based on the tracking

information. This trajectory information can be utilized to analyse the performance of each player.

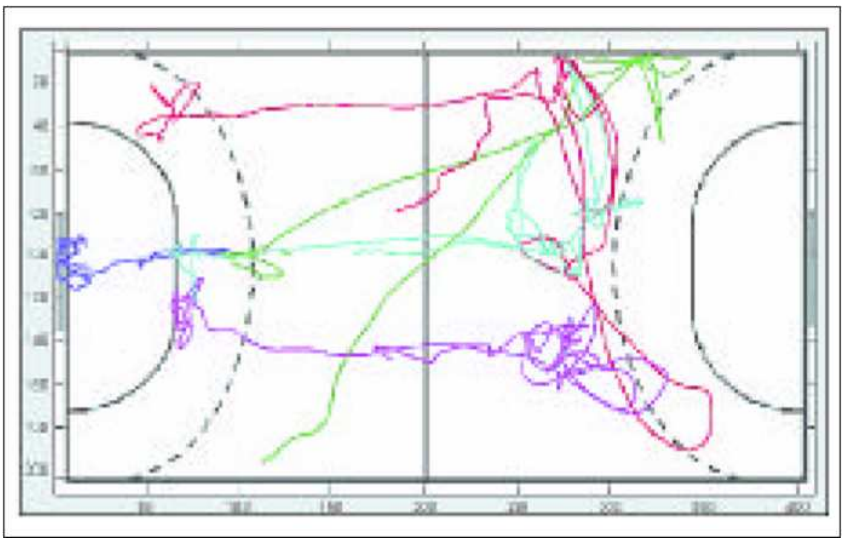


Figure 2-4: The trajectories plotted by the handball players tracking system.

The limitations of this system are:

- This system is designed to work in a constrained environment including illumination conditions which must remain constant during tracking.
- This system is not fully automatic. A human operator is needed to give the initial positions of the players.

Wedge *et. al.* [39] have developed a football tracking system (Figure 2-5). In this system, the trajectory of a ball moving over a cluttered

background is found in panning and zooming camera video sequences. The sequences used in this system are the digital broadcast videos of Australian Football League matches.

Figure 2-5: The football tracking system proposed by Wedge *et. Al* [39].

The football is detected using two methods. Firstly, the football is detected by an adjacent frame difference algorithm. Two consecutive greyscale images are aligned and one is subtracted from the other to search for motion regions. These regions are usually triggered by the motion of a football. Its position will be treated as the start point of a trajectory and is feed into a Kalman filter to predict the location of appearance of this target in the succeeding frame. Secondly, a random sampling algorithm is implemented in this project to choose image pixels based on a Gaussian distribution function around the predicted

position in the current frame. These pixels are compared on the hue channel in HSV colour space to classify the pixels that represent the football and the pixels that represent the background.

A Kalman filter is implemented in this research to predict the trajectory of a football. The football's trajectory is predicted by a 4 dimensional vector: $(x, y, \dot{x}, \dot{y})$, where (x, y) represents the position of the football in the last frame and $(\dot{x}, \dot{y})$ represents the velocity of the football in the last frame. The motion of a football is modelled as a constant velocity motion as the authors believe that the effect of gravity and wind is negligible within the temporal space of successive frames.

One limitation in this project is that the motion model of a football in this system does not work well in the case that the ball bounces on the ground.

Human-Computer Interaction Systems

Computer vision based tracking techniques are also used in human-computer interface design. For example, Iannizzotto *et. al.* [67] proposed a computer vision based fingertip tracking system, the VisualGlove (Figure 2-6), to help people interactive with wearable computers without traditional pointing devices.

Figure 2-6: The VisualGlove, a computer vision based human-computer interaction system [67].

The VisualGlove system detects the tips of both thumb and index fingers by Hausdorff Distance (HD), a computer vision based pattern matching algorithm. The fingertips' positions are tracked by Kalman filter. This system can also detect basic operations, such as mouse click (contact between the two fingers) and drag (keep the index finger and thumb joined and moving the hand at the same time).

The limitations of this novel HCI system are:

- The tracking accuracy degrades rapidly when this system is used in a cluttered environment.
- Tracking is not robust to finger occlusion.
- The accuracy of the system is limited by the resolution of captured images.

Entertainment System

Computer vision based tracking techniques have been recently utilized in entertainment systems. Freeman *et. al.* [68] believes that computer vision techniques allow game players to experience a more natural interaction and engagement in the game.

Vaghani and Green [48] have proposed a computer vision based human body tracking system (Figure 2-7) designed for entertainment purposes with unconstrained clothing and movement. In this "The Lord of the Rings" system, a human body is described as vector of following dimensions:

- Centre of mass
- Principal axis
- Torso positions
- Head centre of mass
- Shoulder positions

A stereo vision system is utilized to segment the human participant in the system and illustrates her or his body motion by animations of a 3D character (Figure 2-7).

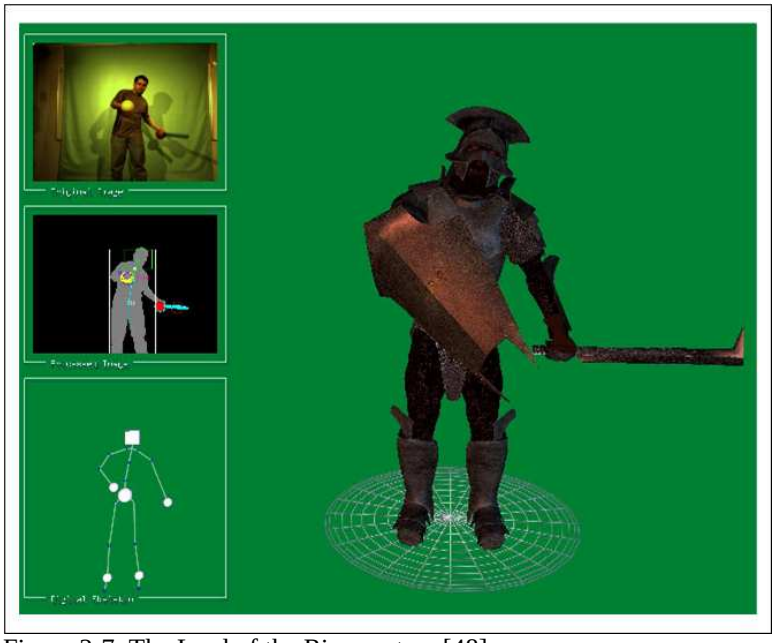


Figure 2-7: The Lord of the Ring system [48].

The major limitation of this system is:

- A constrained environment is required in that a human participant is asked to stand in front of a green screen with constant studio illumination.

In addition to the above computer vision research, the next 2 sections present object detection techniques and target tracking techniques, which will be discussed in more detail.

2.3 Object Detection Techniques

Object detection techniques are used in the computer vision research to segment the targets from a static background in the observed images. The foreground detection techniques in the literature can be classified into 2 categories:

- Stochastic based object detection techniques
- Temporal based object detection techniques

Stochastic based object detection techniques segment the target by means of modelling the background scene. Usually, a stochastic based object detection algorithm contains two steps: a background learning step and a foreground detection step. In the first step, images that contain the pure background scene are feed to the systems that utilize stochastic background modelling algorithms. The stochastic background modelling algorithm models the pure background images pixel by pixel and keeps the models in the system. In the second step, the algorithm conducts a stochastic test over each pixel in the image with the corresponding model it keeps. The pixels that do not match the background models will be segmented as foreground objects. A detailed discussion of this technique is provided in section 2.3.1.

Temporal based object detection techniques detect objects based on the temporal differences between image frames. The algorithms that

belong to this class segment images by their motion features. Usually, the static part of an image is regarded as background and the moving parts of the image are segmented as objects of interest. The temporal based object detection algorithms and their applications are introduced in section 2.3.2.

2.3.1 Stochastic based Object Detection Techniques

Gaussian Background Model

In a stochastic based object detection system, the characters of each individual pixel in the image are modelled by a stochastic description. This description is updated dynamically during image processing. A foreground object can be segmented from the background as their pixels have different characteristics from the background model.

Most of the stochastic based object detection systems [5] [13] [14] [15] model each pixel in the background image as a Gaussian probability distribution function (Equation 2-1).

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/(2\sigma^2)}$$

Equation 2-1: The Gaussian probability distribution equation [42]. In this equation, x is a random variable; μ is the mean value and σ is the population variance

The foreground object detection is done by setting the threshold of the difference between an observed pixel intensity value and the corresponding pixel description in the background model. Suppose the observed pixel p has intensity value $I_p^{'}$; the corresponding pixel's background model is $I_p \sim (\mu_p, \sigma_p)$. Then, the pixel p is classified by the test: $\left| I_p^{'} - \mu_p \right| > Thd$, where *Thd* can be chosen arbitrarily <check all spelling in this thesis>. The k value (usually 3) is the confidence interval.

The background models are updated over time with a running average algorithm (Equation 2-2):

$$\mu_{t+1} = \alpha I_t + (1-\sigma)\mu_t$$
$$\sigma^2_{t+1} = \alpha (I_t - \mu_t)^2 + (1-\alpha)\sigma^2_t$$

Equation 2-2: the running average update

In equation 2-2, α is known as the learning rate. It indicates the model updating speed. The range of the α value is: $\alpha \in [0,1]$. When α equals to 0, the model stops updating and always use the background models that the system currently has. In this case, the value of background models are reduced when the properties of the background scene changes and therefore, the system is prone to false positive errors. On the other end, when the α = 1, the background is updated totally for every frame and therefore the background model of the pixel p can be

expressed as $p_t \sim (I_t, 0)$. In this case, the stochastic based approach fades into the adjacent frame difference algorithm, which will be introduced later. Finally, if α is a particular value between 0 and 1, the background model will update at a rate relative to the α value.

Mixture of Gaussian Background Model

In addition to the Gaussian model introduced above, some stochastic based object detection systems in the literature model the background characters with multiple Gaussian probability distribution functions, for example [7] [43] [44] [45] [46] [50]. The multi-Gaussian model approaches model the background with multiple (usually 3 to 5) Gaussian models. Each model represents one particular case of the background pixel (Figure 2-8). These Gaussian models have different importance to the final models. The multi-Gaussian model can be represented by equation 2-3:

$$P(x_t) = \sum_{j=1}^{K} \frac{\omega_j}{(2\pi)^{\frac{d}{2}} \left|\sum_j\right|^{\frac{1}{2}}} e^{-\frac{1}{2}(x_t-\mu_j)^T \sum_j^{-1} (x_t-\mu_j)}$$

Equation 2-3: A mixture of Gaussian probability distribution.

The value ω_j in the equation 2-3 represents the importance of the j^{th} Gaussian model to the whole mixture and the $\sum_j = \sigma_j^2 I$ is the covariance of j^{th} Gaussian distribution.

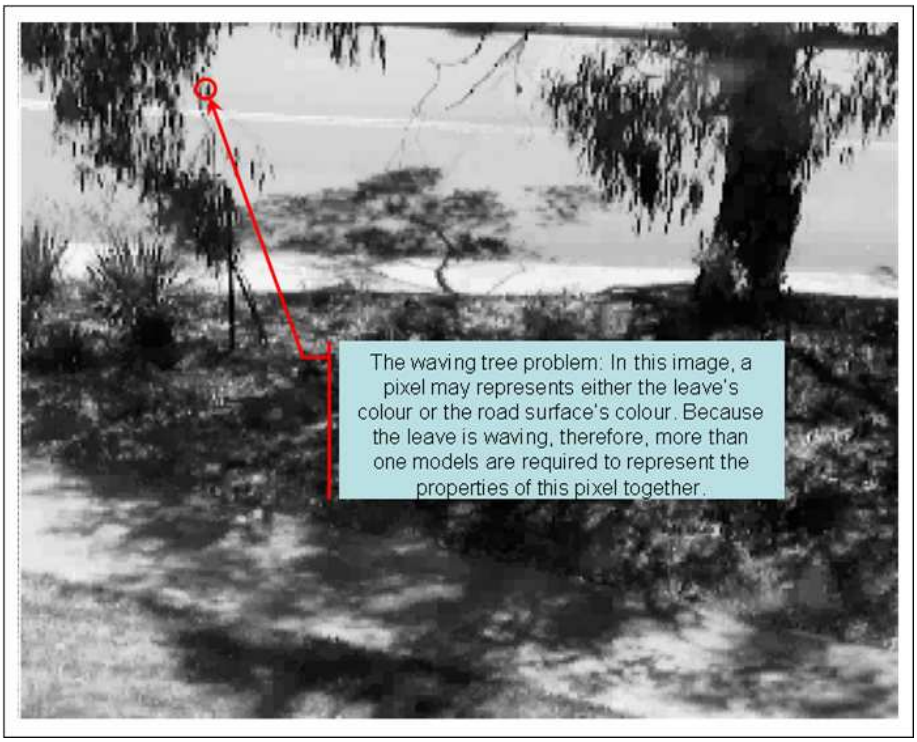


Figure 2-8: The waving tree problem. In this case, a pixel may represent either tree leave or ground surface. Therefore, multiple models are used to represent its properties [44].

Discussion:

The advantages of stochastic based approaches are:

- As the background model is usually established by pure background images during a training period, the foreground objects can be segmented by other than just its contour.

- The background model update step helps these algorithms to gradually adapt to illumination change.
- Stochastic algorithms that use a mixture of Gaussian probability distribution for pixels' models can minimize problems like waving trees and changing shadows [3].

However, the stochastic based approach has following disadvantages:

- To model every pixel in an image is a computationally intensive task.
- A dedicated training period is required during the background modelling stage.
- The background model cannot keep up with the background variation when it changes suddenly.
- As each pixel has its own background model, the camera has to be very stable in stochastic based systems; otherwise the background model has to be recalculated.

2.3.2 Temporal based Object Detection Techniques

Reference Frame Difference

Reference frame difference is a temporal based object detection technique that detects foreground objects in an image by finding the difference between an image and a reference image. Usually, the first

frame (Figure 2-9 [a]) is specified as the reference in the reference frame difference system, such as [47] [48]. Suppose $\boldsymbol{I}_0$ represents the reference image, and $\boldsymbol{I}_t$ represents the current frame (Figure 2-9 [b]). Then the difference frame *D* (Figure 2-9 [c]) can be represented by equation 2-4:

$$D = \left| \boldsymbol{I}_t - \boldsymbol{I}_0 \right|$$

Equation 2-4: Reference frame difference.

The difference frame D is subjected to a threshold value *Thd* and then the binary foreground objects mask can be generated. This mask is a bi-classifier that segments images as foreground and background.

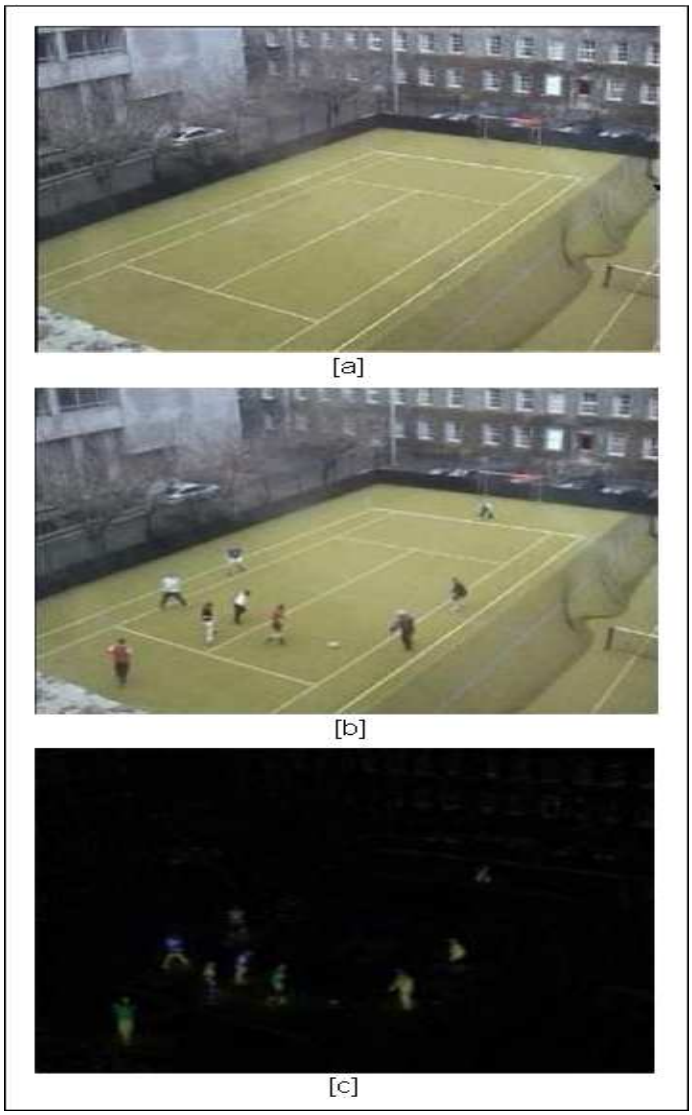


Figure 2-9: The reference image difference algorithm. [a] The first frame that contains pure background image. [b] The current image that contains foreground objects and background. [c] The difference image that represents the detected foreground objects.

Adjacent Frame Difference

The adjacent frame difference is a special reference frame difference algorithm in which the reference frame is always the nearest proceeding frame (Figure 2-10 [a] [b]). The adjacent frame difference algorithm is widely used in the literature to detect moving objects [49] [50] [51]. A difference frame D (Figure 2-10 [c]) can be represented as:

$$D = \left| I_t - I_{t-1} \right|$$

Equation 2-5: Adjacent frame difference

The difference frame D is thresholded with a value *Thd* and then the binary moving objects mask B_{motion} can be generated. This mask is a bi-classifier that segments images into moving objects and a static background.

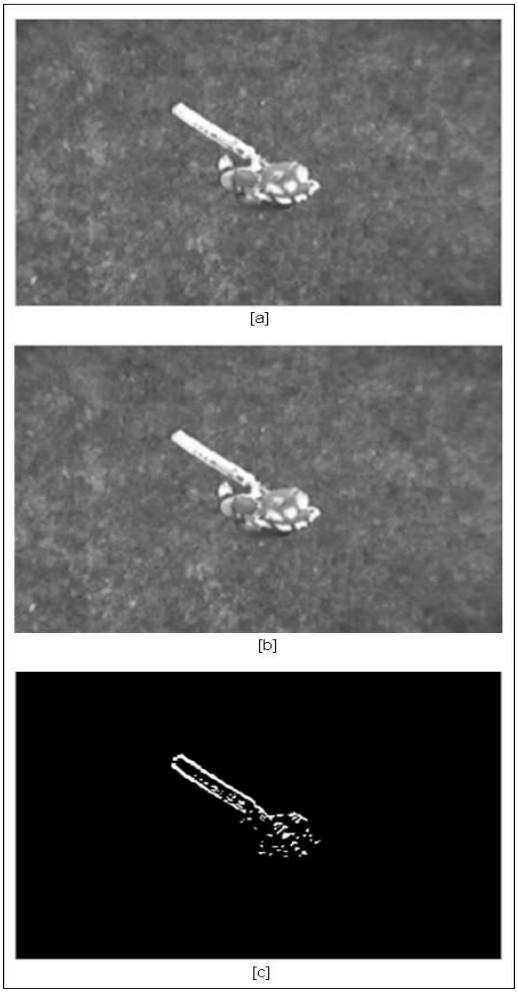


Figure 2-10: The adjacent frames difference algorithm. [a] The reference frame. [b] The current frame. [c] The difference image.

Double Difference Algorithm

The double difference algorithm (DDA) is an improved frame difference algorithm. Instead of computing the difference image of two adjacent frames, it performs a binary "AND" operation over the two binary difference images (Figure 2-11 [f] [g]), calculated using equation 2-5 with 3 adjacent frames (Figure 2-11 [a] [b] [c]). DDA is found in many moving object detection systems in the literature <for example is inferred – remove all equivalent occurrences in this thesis> [1] [2] [4] [6] [24] [26] [27]. Equation 2-6 describes DDA mathematically:

$$D_{i,i-1} = Threshold(|I_i - I_{i-1}|)$$
$$D_{i+1,i} = Threshold(|I_{i+1} - I_i|)$$
$$C_i = D_{i,i-1} \cap D_{i+1,i}$$

Equation 2-6: The double difference algorithm

In equation 2-6, the two binary difference images, $D_{i,i-1}$ (Figure 2-4 [f]) and $D_{i,i+1}$ (Figure 2-4 [g]) are calculated by thresholding two <all numbers, below 20, as part of a sentence should be a word not a digit – change all occurrences in this thesis> difference images (Figure 2-4 [d] [e]) from three adjacent original images [a], [b] and [c]. The DDA binary mask (Figure 2-4 [h]), C_i is then generated by a binary "AND" of the two binary adjacent difference frames.

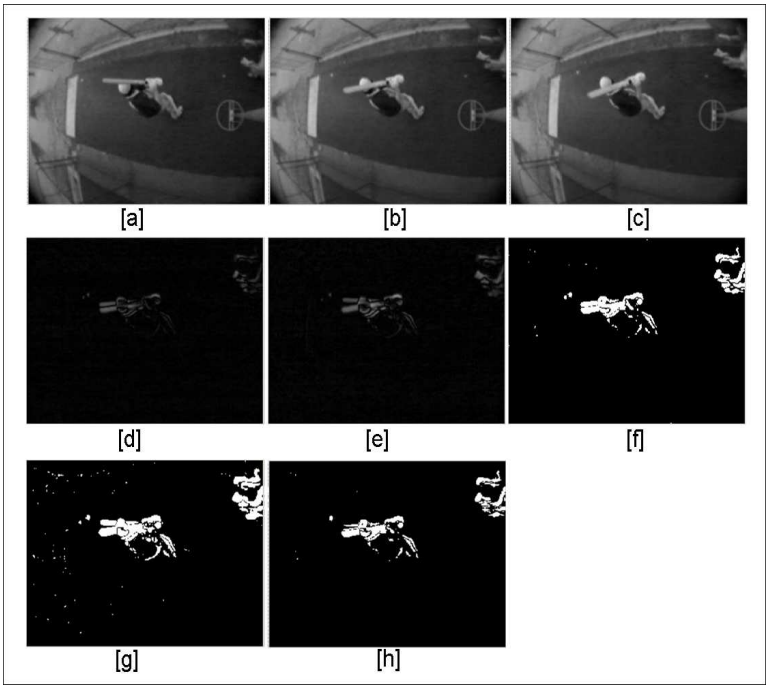


Figure 2-11: The intermediate outputs of the DDA algorithm. [a] – [c]: The original images. [d][e] The two adjacent difference frames. [f][g]: Binary masks generated by thresholding adjacent difference frames. [h] The DDA motion mask.

Discussion:

The temporal based approaches analyse the difference between frames to achieve the goal of object detection. The reference frame difference algorithm uses the first frame as its reference image; the foreground objects are those pixels in the following frames that have a different appearance from the pixel in the first frame. The advantage of this approach is that it can detect the foreground objects that stop in the image. However, as the frame number between the current frame and the reference frame is incremental, the noise remaining in the after-image will be accumulated and therefore the quality of after-image is reduced. Usually, a strict environment control is required when utilizing this approach.

The adjacent frame difference approach detects foreground objects by analysing the difference between the current frame and the frame adjacent to it. As the distance between two frames are constant, the noise in the after-image will not be accumulated. Furthermore, as only the latest adjacent frame is used as a reference frame, the noise triggered by sudden environmental variations will be limited to just a few frames. On the other hand, the limitation of this approach is that it can be used to detect moving objects only. A static foreground object will be ignored by this approach. Furthermore, if a moving object appears at two adjacent frames, the difference image will contain two areas that indicate the target position but only one of them is the correct representation. This limitation is referred as the ghost object problem [3].

The double difference approach is an improved adjacent frames difference algorithm. It solves the ghost object problem. Furthermore, as an extra frame is involved, the noise of the resulted image is reduced. As with the adjacent frame difference approach, this algorithm can only be used to detect moving objects.

2.4 Target Tracking Techniques

Tracking is a process that matches objects' features, such as colour, edge, contour and blob, in consecutive images. Although the matching

operation can be done in the whole image, usually a region of interest (ROI) is defined. There are two advantages in matching target's features in the ROI:

- Improve processing speed. As the search space is pruned, the pixels that need to be processed in the ROI are less than that of the whole image. Therefore, the processing speed will increase when ROI is defined.
- Reduce the error. There may be many areas in an image that have similar features as the tracking objects. By defining the ROI, the affect of these areas on the matching algorithm may be reduced. Therefore, the matching error can also be reduced.

In the literature, two major categories of approaches are used to infer the ROI in the succeeding frame. Firstly, define the ROI as the area that surrounds the target location observed in the current frame [47]. Secondly, utilize a stochastic approach to estimate the potential location that the target will appear in the succeeding frame.

The advantage of the first approach is its simplicity. However, in these systems, the motion properties of the target object has to be fairly simple and stable, otherwise, as illustrated in the figure 2-12 [b] [55], the tracking is lost because the target exceeds the predefined ROI area.

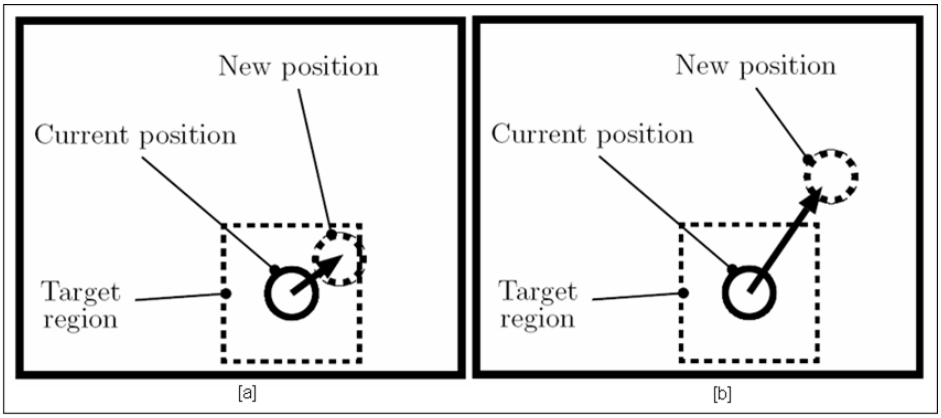


Figure 2-12: Tracking object without position prediction.[55]. [a] The target appears in the ROI area of the image; the tracking is successful in this case. [b] The target appears out of the ROI area of the image; the tracking is unsuccessful in this case.

To overcome the shortcomings that exist in the first approach, the location of a target in the succeeding frame is inferred by a stochastic estimation in the second approach. In such systems, the tracking process is defined as “the problem of generating an inference about the motion of an object given a sequence of images” [53]. In other words, tracking can be defined as the process of determining the location of a particular feature, for example colour, in an image sequence over time [40] [55].

A Kalman filter [32] is a stochastic approach to predict the location of the tracking object in the succeeding frames. It has been widely used in computer vision based tracking systems, such as [52] [56][57]<don’t range refs> - 63]. Kalman filter is a recursive data processing algorithm that infers the variable of interest based on the observed variables in the past (the measurement data) and prior knowledge of the system [54]. A Kalman filter contains two stages in

each iteration step; they are the prediction stage and the measurement stage. The prediction stage, also referred to as the time update stage, is responsible for projecting forward in time the current state and error covariance estimators to obtain an estimate of the next time step. The measurement stage, on the other hand, incorporates a new measurement into the previous estimation (from the prediction stage) to obtain an improved posterior estimate [31]. In other words, the prediction stage predicts the target state in the next frame and the measurement stage corrects the estimation by observed state from the succeeding frame. The Kalman filer can be depicted in figure 2-13 [31].

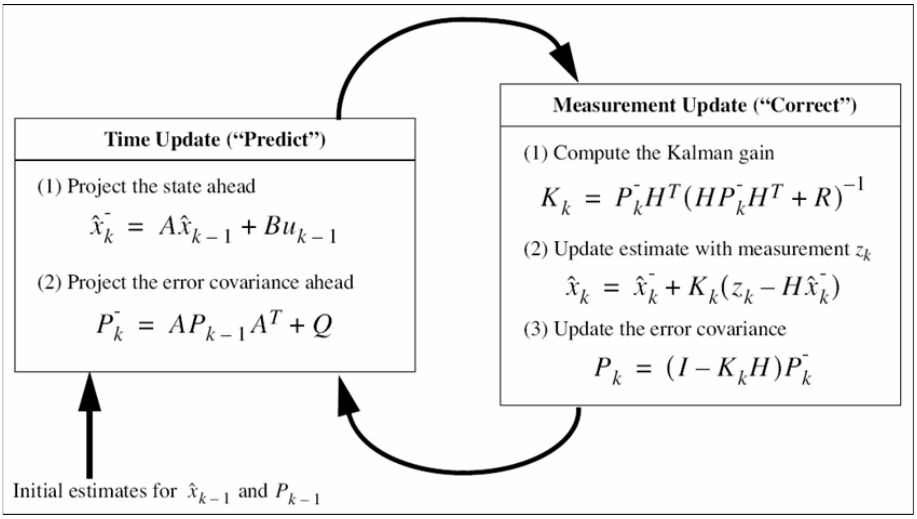


Figure 2-13: A complete picture of the Kalman filter operation [31].

2.5 Conclusion

In this chapter, computer vision systems are discussed briefly with specific focus on object detection and target tracking techniques in

recent literature, which are discussed in section 2.3 and section 2.4 respectively.

The computer vision research introduced in this chapter can be classified into four categories based on their goals:

- Video surveillance systems
- Sport training systems
- Human-computer interaction systems
- Entertainment systems

Stochastic approaches and temporal approaches are the commonly used techniques to detect foreground objects in a video sequence. The stochastic based approaches detect the foreground objects by finding the difference between the visual properties in the foreground objects' pixels together with statistical models of the corresponding pixels in the background. The temporal approaches detect foreground objects by finding the difference between the current frame and a reference frame.

The computer vision based target tracking necessitates matching similar features in a sequence of images. The matching operation is usually done in the ROI of the seeking image. A Kalman filters are commonly used in the literature to predict the ROI during the tracking operations.

Chapter 3

Project Overview

3.1 Introduction

In this chapter, the research motivation is described in section 3.2. The reasons for testing algorithms using a ball tracking system are explained in section 3.3. The major components of this tracking test environment are explored in section 3.4.

3.2 Research Motivation

The research goal of this project is: To overcome the limitations of prior research and develop a novel computer vision algorithms which can detect and track small and fast objects in low quality image sequences.

The proposed algorithms should be able to:

1. Correctly detect small diameter objects in a relative large search space correctly.
2. Correctly track moving objects even when the targets' appearances vary from frame-to-frame.
3. Detect and track fast objects
4. Keep tracking through large trajectory discontinuities.

3.3 The Tracking System

As described in the last section, the motivation of this research is designing computer vision algorithms that can detect small objects successfully and track fast targets robustly in low quality image sequences. We propose a cricket ball tracking system that fulfils the above requirements raised by the research motivation. In this section, we will introduce this cricket ball tracking system in brief and then explain why this cricket ball tracking system is selected to test the research approach.

In this proposed ball tracking system, a wireless camera placed above the batter is used as the video acquisition device for this system. The camera is tilted 34 degrees forward of vertical (facing down) to get the maximum field of view. A computer is used to receive and analyse image sequences that are captured by the wireless camera to detect and track cricket balls both inward and outward directions.

The cricket ball tracking system provides an experimental environment for:

1. Detecting small diameter objects (the cricket balls) in a relatively large search space.

   Figure 3-1 illustrates an image that is captured by the camera. Due to the lens distortion, the actual view field is trapezoid in shape (Figure 3-2). The size of this trapezium is approximately 56 square meters. In this setup, a cricket ball may appear as only 1 or 2 pixels (worst case) in the image after being segmented from the background (Figure 3-3).

Figure 3-1: A frame captured by the camera in the cricket ball tracking system.

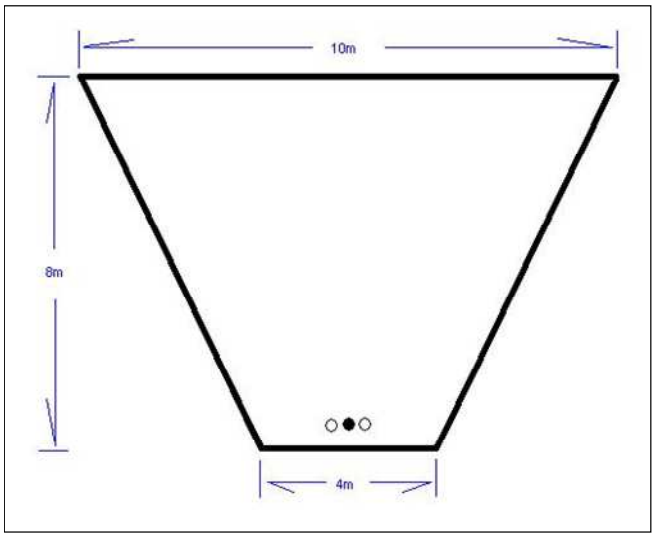


Figure 3-3: The trapezium field of view.

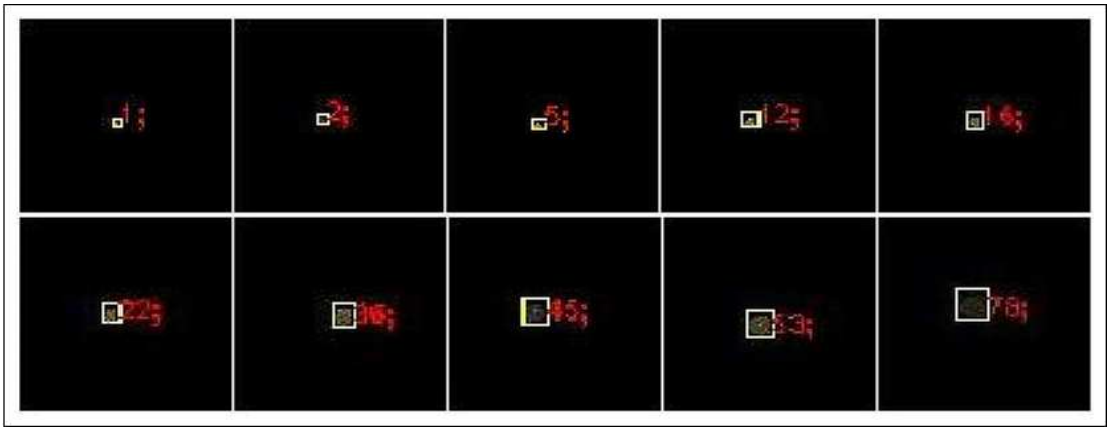


Figure 3-3: The size of a cricket ball in the image. The number in each picture indicates the number of pixels that a cricket ball occupies. The size of the cricket ball in this tracking system may vary from a minimum of 1 pixel to about a maximum of 80 pixels.

2. Tracking targets while their appearances vary dramatically from frame-to-frame.

   As demonstrated in figure 3-3 and figure 3-4, a cricket ball's shape and size varies from frame-to-frame dramatically because of

   - Variance of the lighting condition.
   - Similarity of the background colours and textures
   - The distance between the camera and the target.

   In addition, the cricket ball's colour is also varies to the observers due to:

   - The inherent limitations of a CCD camera under various illumination levels.
   - Signal noise arising from the power supply, background radio signal noise and radio transmission distance.

   For example, in figure 3-5, the colour images fade into gray scale images due to the reasons mentioned above.

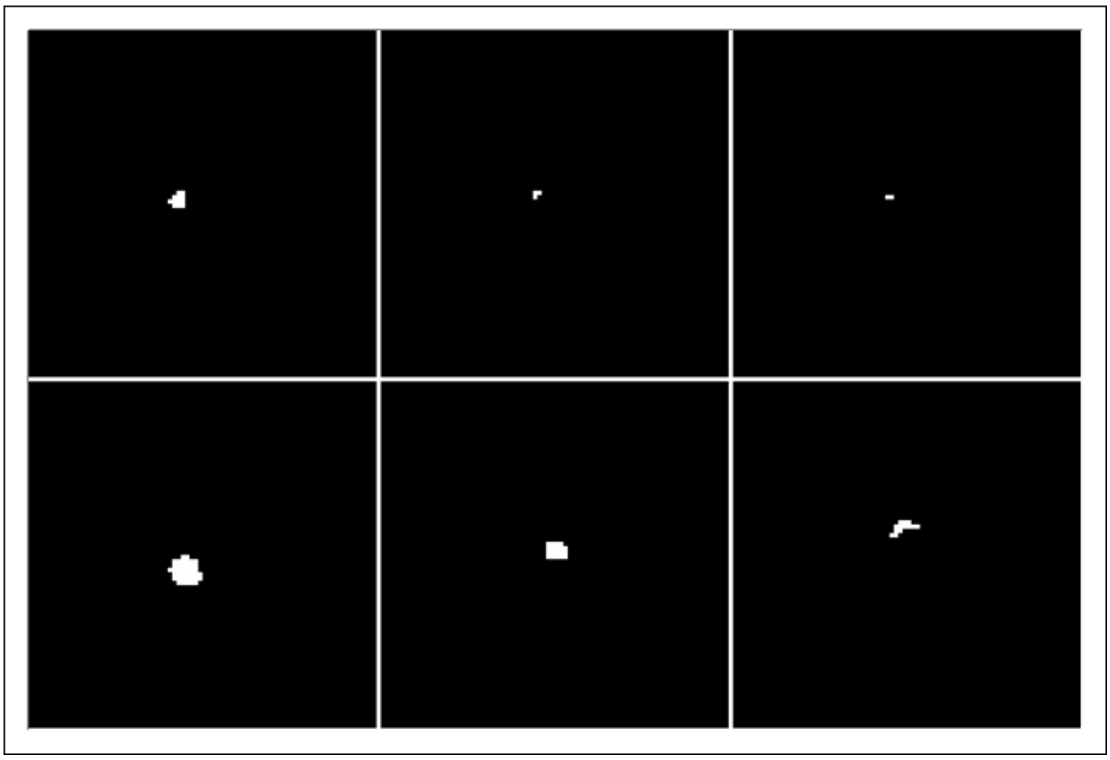


Figure 3-4: The shape of a cricket ball varies dramatically after motion image segmentation procedure. The tracking system should be able to track cricket balls with different shapes properly.

Figure 3-5: These four images are captured from the same video clip. The first and last frames are in colour but the two frames in the middle have minimal colour content - almost gray scale images <label frames a,b,c,d>.

3. Detecting and tracking fast objects.

   In this proposed cricket ball tracking system, the maximum speed of a cricket may be as fast as 150km/hr. The reliability of the object detection and target tracking algorithms can be evaluated with fast moving targets in this system.

4. Keeping tracking through large trajectory discontinuities.

   The trajectory discontinuity of a cricket ball in the proposed cricket ball tracking system comes from three sources.

   - Radio transmission error
   - Inconsistent background texture inconsistent
   - External forces from physical contact between objects or ground

   First, radio transmission error may involve two problems: missing image frames and image frame quality deterioration. Figure 3-6 illustrates two apparent **adjacent** frames in a sample video footage. Dropped frames that should exist between these two images are missing because of a data transmission error. In addition, radio transmission errors may also arise from a wide range of image noise and distortion, for example figure 3-7. Tracking targets may not be able to be reliably detected in such low quality image sequences.

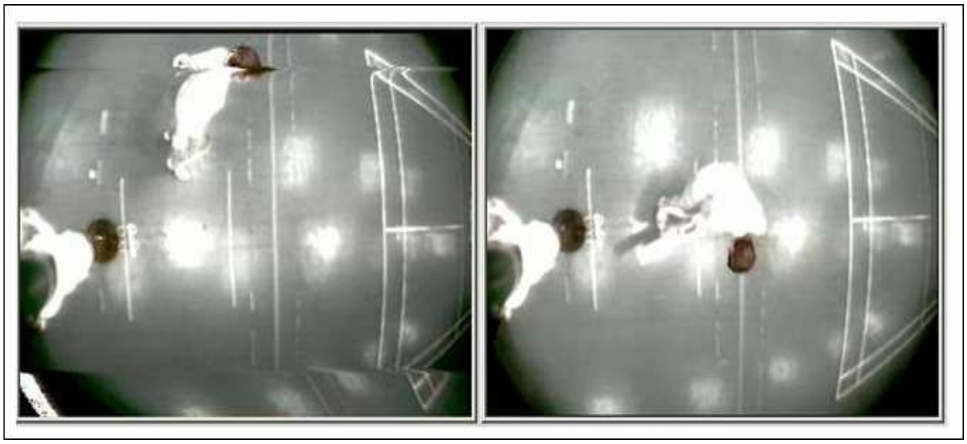


Figure 3-6: The images illustrate two adjoining frames in one sample video. The batter' positions in two frames are disconnected, which indicates that the information between these two frames has been lost.

Second, as the ground texture is inconsistent in the real environments, a ball may merge into the background in some areas where the background has a similar colour to a cricket ball. For example, in figure 3-8, the ball hue matches a light gray colour over the concrete surface. Background surface reflection can also affect ball contrast. In figure 3-9, the ball appears to disappear over a bright background area.

Finally, a cricket ball's trajectory may change dramatically when external forces are applied. For example, in figure 3-10, the cricket ball bounces on the ground (Figure 3-10) with a consequent trajectory discontinuity. Another case that involves large discontinuities in a cricket ball's trajectory is when the batter hits the cricket ball. The cricket ball may then change direction significantly in a very short time.

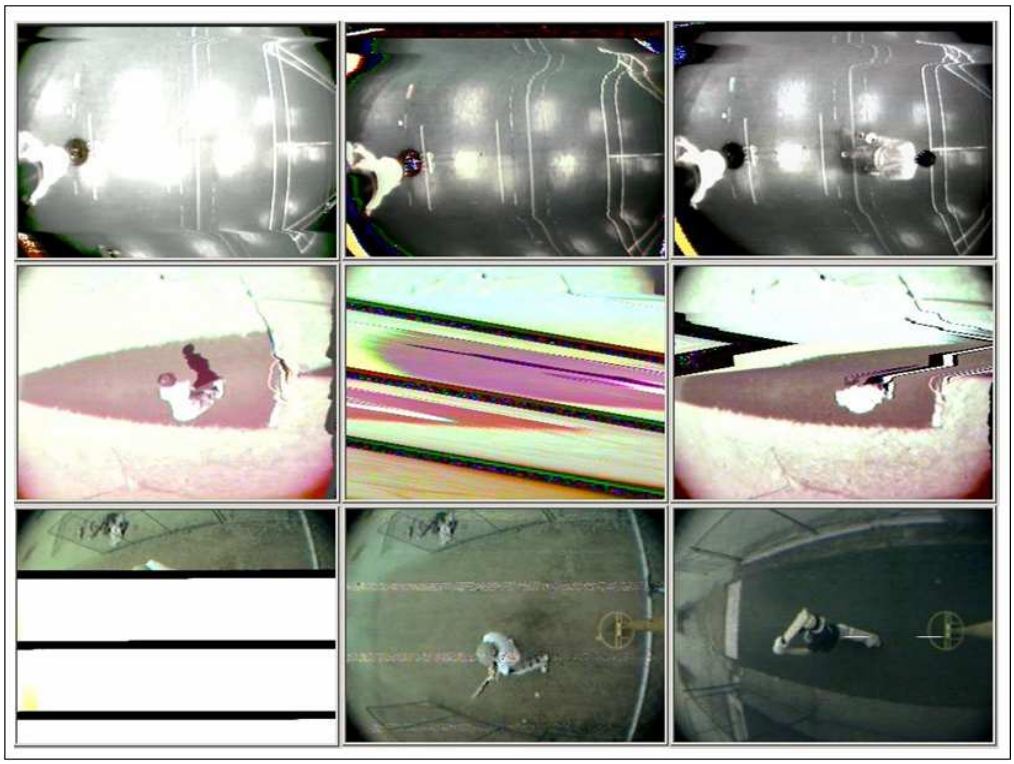


Figure 3-7: The quality of image is reduced due to radio transmission/reception errors.

Figure 3-8: These figures show two examples of the cricket balls merging into the hue of a concrete region and becoming visually indistinguishable from the background.

Figure 3-9: The six images above are adjoining frames in a sample video clip. In the two frames in the middle, the cricket ball disappears in the blue rectangular area because of the ground reflection.

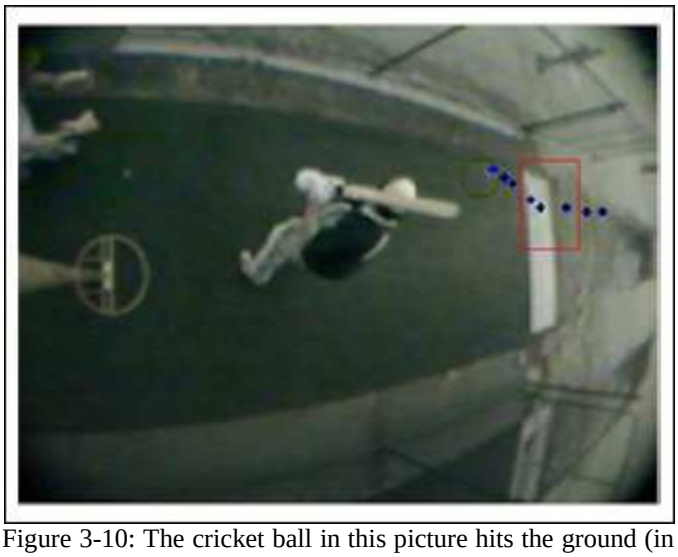


Figure 3-10: The cricket ball in this picture hits the ground (in red rectangular area) and its trajectory is changed.

In a summation, the proposed ball tracking system fulfils the requirements to test algorithms tracking small, fast noisy objects as mentioned in the research motivation section.

3.4 Components of the tracking system

In this section, the major components of the video capturing subsystem, object detection subsystem, target tracking subsystem and trajectory information visualization subsystem will be briefly introduced. Detailed design and implementation information will be discussed in the next 4 chapters.

The video capture subsystem provides image sequences that contain cricket ball motion information. In this system, a wireless camera, radio receiver and frame grabber are used to capture image sequences at frame rate of 30 frames per second. Detailed description of the video capture subsystem is in chapter 6.

The object detection subsystem extracts the moving target from the static background. This subsystem feeds positional information of a moving object to the target-tracking module. Both stochastic approaches and temporal approaches are used for motion segmentation. This subsystem is discussed in detail in chapter 4.

The target tracking subsystem tracks the cricket ball tagging its motion features. This subsystem analyses the moving objects that are provided by the object detection subsystem and keeps track of the target from frame to frame. This subsystem also records all key positions along the target' trajectory and provides its trajectory

information to the visualization subsystem. The tracking subsystem is discussed in chapter 5.

The trajectory visualization subsystem plots the cricket ball's trajectory on the screen. The visual trajectory can be used to evaluate the object detection and the target tracking algorithms. The implementation of this subsystem is discussed in chapter 6.

3.5 Conclusion

The motivation for this study is to overcome the limitations of prior research and develop a novel computer vision algorithm which can detect and track small and fast objects in low quality image sequences.

A cricket ball tracking system is proposed in this chapter, which fulfils the requirements detailed in the research motivation.

The proposed cricket tracking system consists of four major components, they are: a video capturing subsystem, an objects detection subsystem, a targets tracking subsystem and a trajectory information visualization subsystem. These components will be discussed in the following chapters in more detail.

Chapter 4

Object Detection Algorithms

4.1 Introduction

Object detection is a very important step in computer vision based target tracking systems. This step classifies the pixels in an image into two categories: candidate pixels of interest and the remaining pixels that the system is not interested in. For example, in this research we are only interested in the objects that have fast velocities. Pixels that describe static background objects in the images should be separated from the pixels of interest.

In this chapter, two novel object detection algorithms, the Hierarchical Object Detection (HOD) algorithm and the Pixel Exclusion Double Difference Algorithm (PEDDA) are introduced and discussed. Both of the algorithms are designed to be robust and achieve a high computational efficiency approach to detecting small, fast, noisy moving objects. The HOD algorithm is a stochastic based object detection algorithm. It detects moving objects by evaluating the stochastic properties of pending image units with the stochastic descriptions of the static background known through background modelling stage. The PEDDA algorithm is a temporal based moving objects detection approach. This algorithm is developed on the basis of general DDA algorithm proposed in [1]. In addition to the advantages achieved by general DDA

algorithms, the PEDDA algorithm has the advantage of suppressing slow moving objects in favour of fast moving objects and is therefore especially useful in this research.

The HOD algorithm is discussed in section 4.2. The DDA algorithm and the PEDDA algorithm will be discussed in the section 4.3. The conclusion for both approaches is shown in section 4.4.

4.2 The Hierarchical Object Detection Algorithm – a Stochastic Approach

The Hierarchical Objects Detection (HOD) algorithm is a novel moving object detection algorithm. The hierarchical structure proposed in this algorithm improves its efficiency by bounding the moving objects detection and background model updating operations into restricted areas. Furthermore, a sensitive area isolation technique is implemented based on this novel hierarchical structure; the algorithm is able to update its background model even when foreground objects exist.

This HOD algorithm is introduced in this chapter. In section 4.2.1, the novel hierarchical structure and the advantages of establishing this hierarchical structure are discussed. In the HOD algorithm, the background models are inferred hierarchically. The background modelling process is discussed in section 4.2.2. In addition, the hierarchical background model updating and foreground objects detection operations are discussed in section 4.2.3 and 4.2.4 respectively. Finally, in section 4.2.5, apart

from summarizing the advantages and disadvantages of this algorithm, the reasons for not using this algorithm in the tracking system implemented in chapter 6 are discussed.

4.2.1 The Hierarchical Structure

The HOD algorithm represents each image as a combination of 5 levels hierarchically. Each level contains a description of the original image at different resolutions. At the coarsest level, a 320 by 240 pixels image (Figure 4-1[a]) is represented as a 20 by 15 matrix of intensity values (Figure 4-1[b]). Each value in this matrix is a Gaussian of the intensity properties of a 16 by 16 block (Macro Block [8]) in the original image (Figure 4-1[c] and [d]). At the second lowest resolution level, the 320 by 240 pixels image is summarized by a 40 by 30 matrix of intensity values. Each value in this matrix represents the intensity properties of an 8 by 8 block in the original image. These representations are similar in the next two levels, which uses a block size of 4 by 4 and 2 by 2 respectively. At the highest resolution level, the whole image is represented by a 320 by 240 matrix. Each value in this matrix is identical to the pixel's intensity value in the original image. In figure 4-2, the hierarchical structure of whole 5 levels is illustrated.

The HOD algorithm achieves following advantages by utilizing the novel hierarchical structure:

- High performance in foreground objects detection.
- Partial background model updating.

These advantages will be discussed in the corresponding sections respectively.

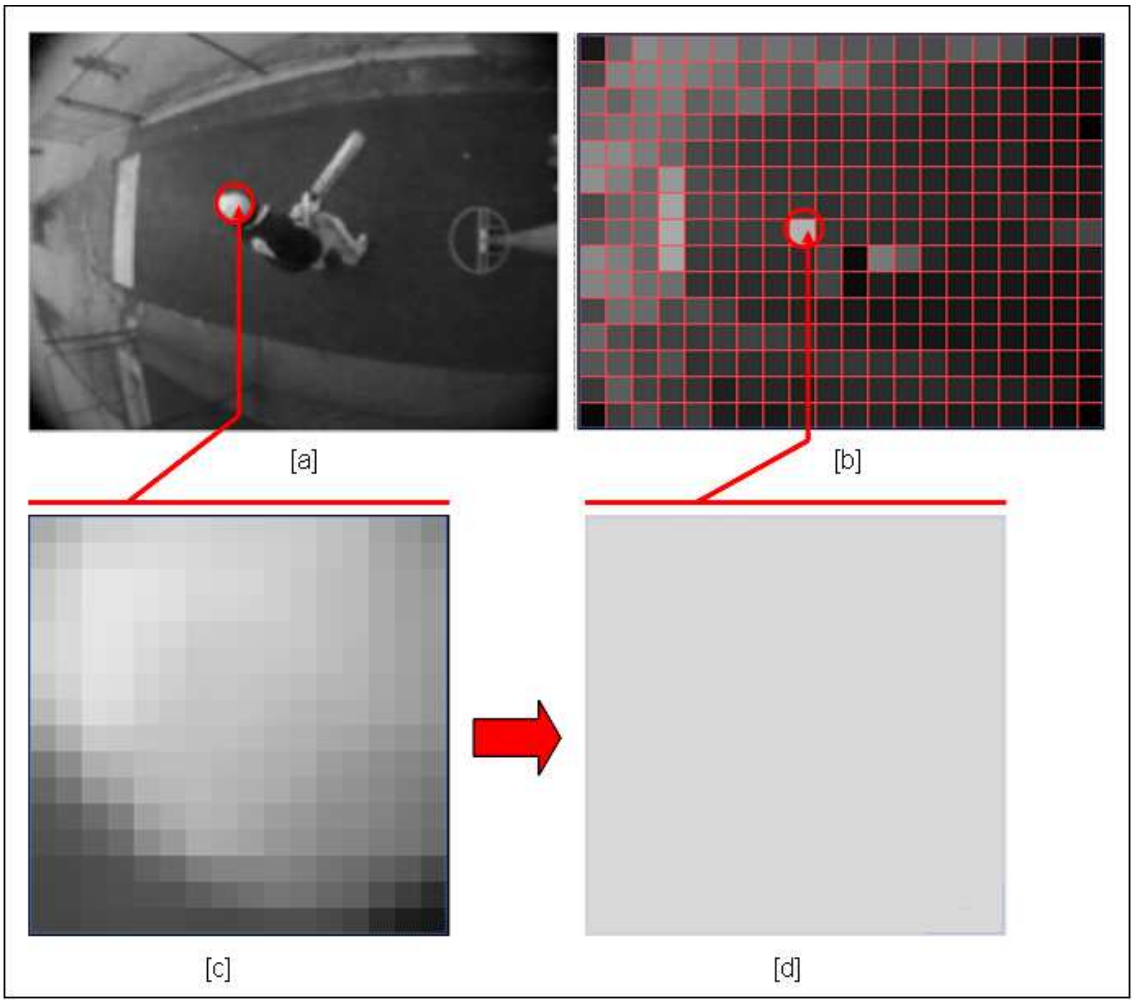


Figure 4-1: Progressive levels of the image representation. [a] a 320 by 240 image [b] the image is represented by a 20 by 15 matrix. [c] a 16 by 16 pixels block in the original image [d] the intensity value that represents the whole 16 by 16 blocks; this value is calculated by the Gaussian of all 256 intensity values in the 16 by16 block.

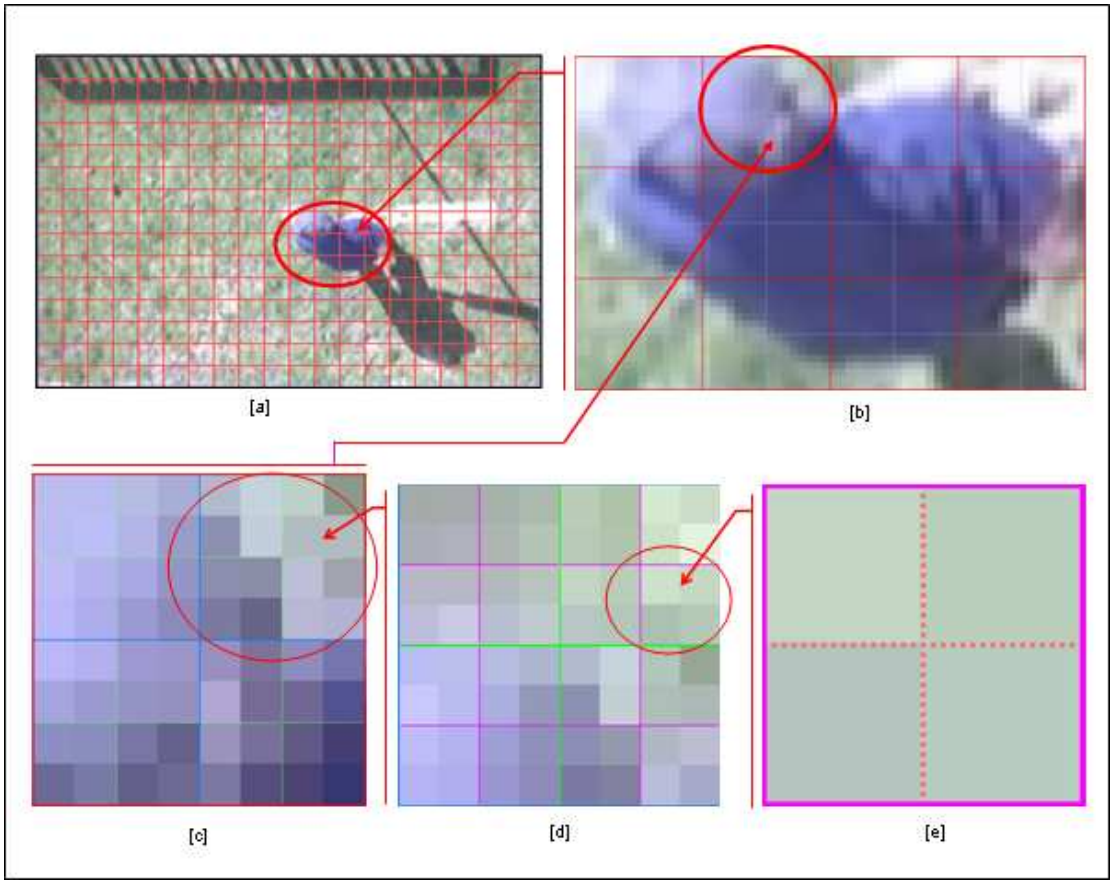


Figure 4-2: The hierarchical structure of the HOD algorithm. In figure [a], a 320 by 240 image is divided into 300 (20 by 15) 16 by 16 blocks. In figure [b], each 16 by 16 block contains four 8 by 8 level blocks. In figure [c], the 8 by 8 block is divided into four 4 by 4 blocks. In figure [d], one 4 by 4 block contains four 2 by 2 blocks. In figure [e], one 2 by 2 block contains 4 pixels.

4.2.2 Background Modelling

Background modelling is a process of establishing a statistical description of an image sequence. To model a series of images, two steps are usually involved: the *background learning step* and the *model fitting step* <perhaps use italics rather than bold in such instances thru this thesis>. In the background learning step the properties of each modelling unit over several images are accumulated by the background modelling algorithm. These properties are fitted into a statistics description in the second step.

Usually, the stochastic background modelling algorithms, such as [7] [12 - 15] [64], establish background descriptions for each pixels. The main disadvantage in these systems, which is raised by pixel level background modelling, is low computational efficiency. As the backgrounds are modelled only in pixel units, the foreground detection operation has to be done on each pixel in each frame. For example, in a 320 by 240 image, 76800 comparisons have to be done for each frame to complete the detection operation. The HOD algorithm relieves the computation complexity of object detection by establishing background models hierarchically. In this section, the modelling process of these hierarchically background models are introduced.

Background Learning Step

As introduced in the section 4.2.1, the HOD algorithm represents the whole image as a combination of 5 levels. All the 5 levels accumulate the properties of background images at the same time in the image learning step.

In the image learning step, the properties of the background images are accumulated in a special data structure: image intensity accumulator (IIA). The IIA (Figure 4-3) is a 3 dimensional data structure. The *x* and *y* dimension in IIA represent the spatial scale of the image representation. For example, at 16 by 16 block level, the *x* and *y* dimension of IIA are 20 and 15 respectively, because the precision of this level is 20 by 15. As each level has its own IIA, the (*x*, *y*) scales of these IIAs are (20,15) at 16 by 16 blocks level, (40, 30) at 8 by 8 blocks level, (80, 60) at 4 by 4 blocks level, (160, 120) at 2 by 2 blocks level and (320, 240) at pixels level <so many numbers - leave as

digits in sentences like this>. The *z* dimension of the IIA represents gray scale intensity values. As 8 bit gray scale images are used in this research, the range of *z* dimension are [0, 255] in IIAs of all levels. For convenience, the three dimensions of an IIA are renamed as *x* dimension, *y* dimension and gray dimension (Figure 4-3).

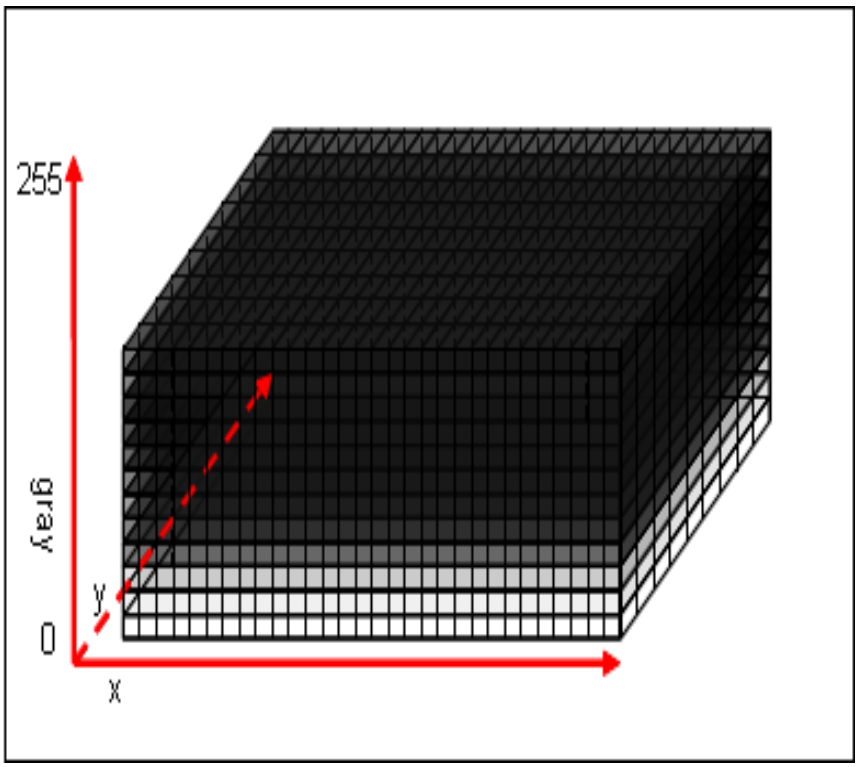


Figure 4-3: IIA, a three dimensional data structure.

Each cell in an IIA can be uniquely indexed by a vector of (x, y, gray). The physical meaning of this index vector is: the intensity value at particular location in an image. The goal of background learning is achieved by counting the number of times that a particular cell has been referenced. The details of this process are illustrated in figure 4-4 using an example of a 16 by 16 block level IIA.

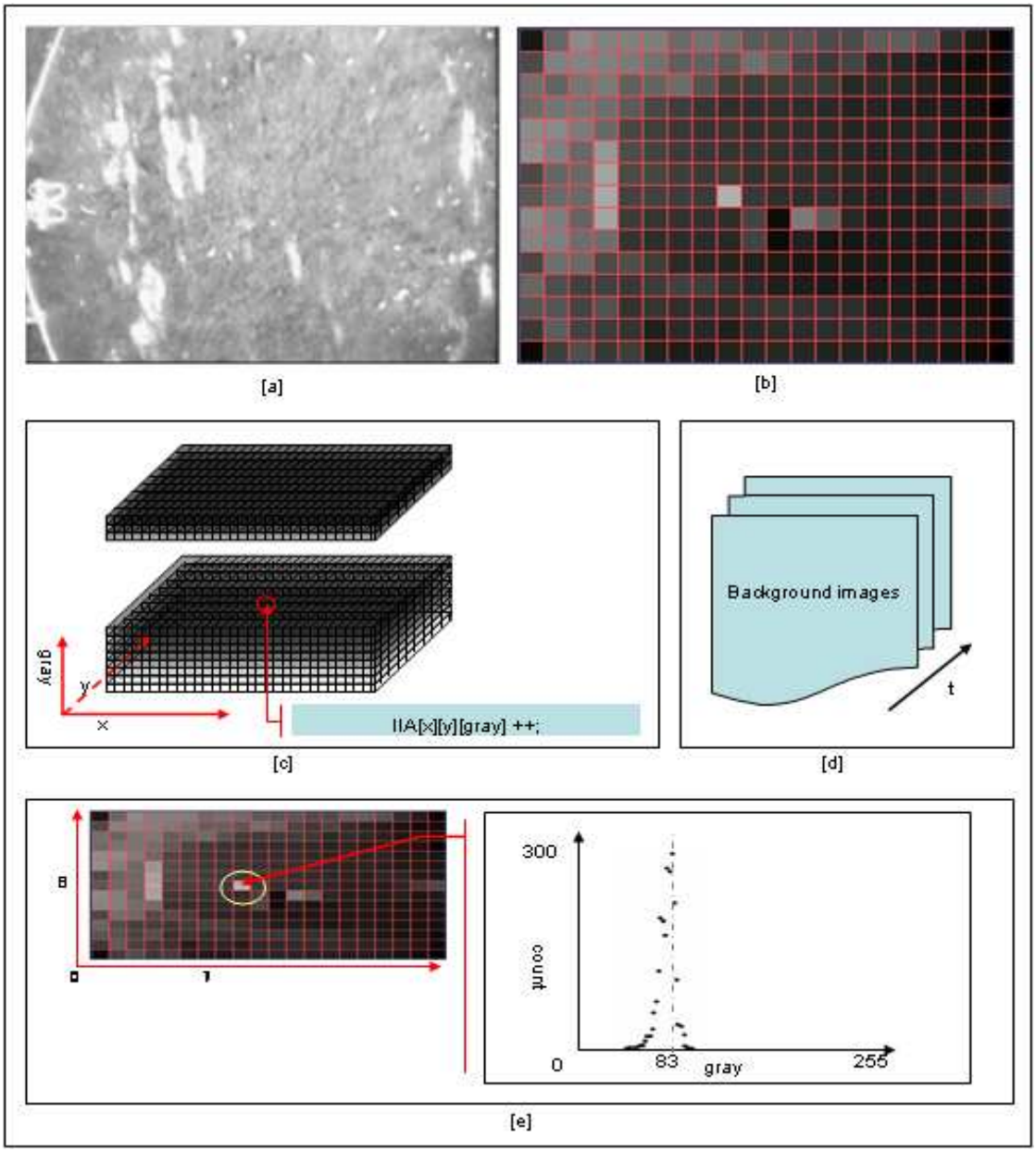


Figure 4-4: An example of background learning in 16 by 16 blocks level IIA. [a] The original image. [b] 16 by 16 block layer gray value. [c] The IIA. [d] Accumulate over time. [e] The accumulated results.

Here the background learning process is explained using an example of a 16 by 16 level IIA. IIAs at other levels work in a similar way as has been introduced here. An original background image is illustrated in the figure 4-4 [a]. This is a 320 by 240 8 bit gray scale image. This original image is summarized by means of every 16 by 16 blocks and these means are stored in a 20 by 15 matrix like figure 4-4 [b]. In other words, figure 4-4 [b] is a low resolution (20 by 15) representation of the original

background image. For each value in this low resolution matrix, an IIA index vector (*x, y, gray*) can be generated by picking up the intensity value (the gray dimension coordinate) in this matrix at location (*x, y*). The cell in IIA that is indexed by (*x, y gray*) is referenced and its counter is increased (Figure 4-4 [c]). This process is repeated for all images in the background learning step (Figure 4-4 [d]). Finally, a description of the properties of the background images is stored in the IIA. For example, at location (7, 8), the most popular intensity is 83, which has been observed 300 times and no intensity values that more than 128 were observed at this location (Figure 4-4 [e]).

Model Fitting Step

The HOD algorithm fits the background image properties accumulated by each IIA into Gaussian probability distributions [16] (Equation 4-1) because of the Central Limited Theorem [17] [18].

$$p(y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y-\mu)^2}{2\sigma^2}}$$

Equation 4-1: the Gaussian probability distribution equation

A Gaussian probability distribution is uniquely defined by two parameters: the mean and the standard deviation. The mean is inferred from unweighted average of the sample values [19] (Equation 4-2) and the standard deviation is inferred by Equation 4-3 [20].

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Equation 4-2: mean

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

Equation 4-3: standard deviation.

To fit the background model of each unit in a Gaussian probability distribution, the accumulated knowledge of background samples in the IIAs are studied. As each level keeps its stochastic description of background independently, the model fitting process is performed over all IIAs concurrently. Here, the 16 by 16 blocks level IIA is utilized again to explain the model fitting step in HOD algorithm. The stochastic inference processes at other image levels are similar to what is introduced here.

As shown in figure 4-4 [e], each unit of the IIA describes the frequency of a particular intensity value observed during the background learning step. The mean value of the intensity values at unit (X, Y) can be calculated using equation 4-4:

$$\mu = \frac{\sum_{g=0}^{g<256}[IIA(X,Y,g) * g]}{\sum_{g=0}^{g<256} IIA(X,Y,g)}$$

Equation 4-4: Calculate mean of the intensity values in unit (X, Y). In this equation, IIA(x, y, g) indicates the values indexed by (x, y, g) in the IIA. The X and Y range from [0, 20] and [0, 15] respectively at 16 by 16 blocks level.

In addition, as the mean is known, the standard deviation of the same sample can be calculated using equation 4-5:

$$N = \sum_{g=0}^{g<256} IIA(X, Y, g)$$

$$\sigma = \sqrt{\sum_{g=0}^{g<256} [(g-\mu)^2 * \frac{IIA(X, Y, g)}{N}]}$$

Equation 4-5: Calculate the standard deviation of the intensity values in unit (X, Y). In this equation, IIA(x, y, g) indicate the values indexed by (x, y, g) in the IIA. The X and Y range from [0, 20] and [0, 15] respectively at 16 by 16 blocks level.

By now, the mean and standard deviation of a particular unit have been inferred from the accumulated background knowledge in IIA. Through this approach, the background descriptions of all 5 levels can be generated.

4.2.3 Background Model Updating

The HOD algorithm updates its background models to keep the stochastic background models and the properties of real world background images consistent. The hierarchical structure of the HOD algorithm isolates the foreground objects in an image from the static parts of the image with 16 by 16 blocks. Therefore, this algorithm is able to update its background model even when the foreground objects are visible. This novel partial background updating algorithm is developed on the basis of selective background updating algorithm in [7].

An active block mask (ABM) is built at the 16 by 16 blocks level (Figure 4-5). The ABM is an indicator that separates the blocks that may contain foreground objects from the blocks that contain the static background image only.

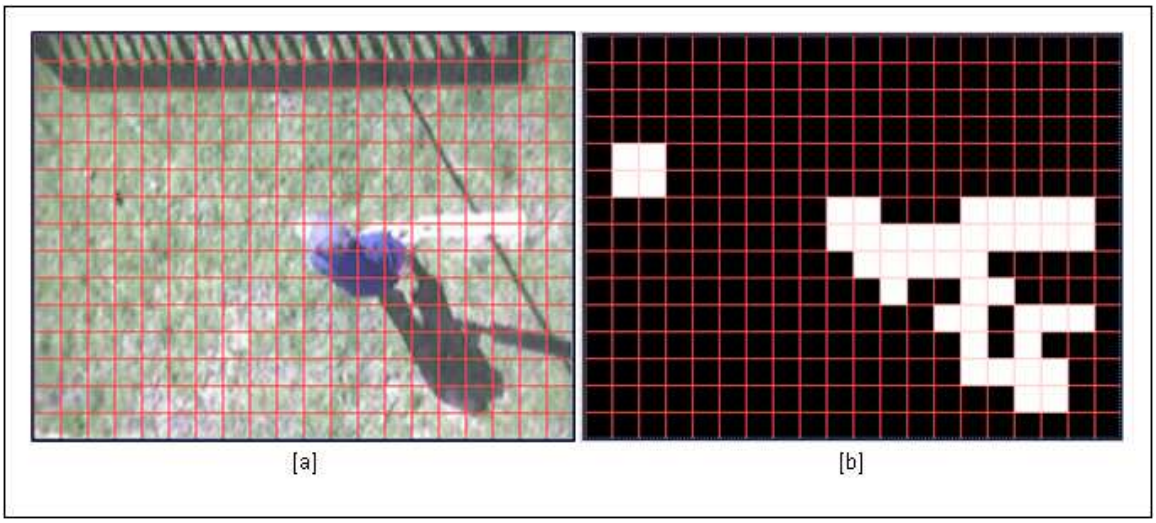


Figure 4-5: [a] the active block mask (ABM) is utilized to separate active blocks (the blocks that may contain foreground objects) and static blocks. [b] In the ABM, the white colour indicates that foreground objects have been detected in these blocks and the black colour indicates that these blocks remain static.

The ABM is only built at the coarsest level. This measurement is designed to reduce the probability of polluting background models of those blocks surrounding the foreground objects. Particularly, the motion-blur effect makes it is very hard to isolate a moving object from the surrounding background accurately at high precision levels.

The background model updating algorithm in HOD checks the ABM before it updates the background models of blocks at any precision levels. If the ABM indicates that foreground objects have been found in the 16 by 16 block that contains the current block, the background model of this block will not be updated in this frame.

The run length background model updating algorithm proposed in [21] [22] [23] is implemented in this HOD algorithm. The two descriptors of a background model, mean and standard deviation are updated by equation 4-6 and equation 4-7 respectively.

$$\mu_{t+1} = \alpha \times I_t + (1-\alpha) \times \mu_t$$

Equation 4-6: run length mean

$$\sigma_{t+1} = \sqrt{\alpha \times (I_t - \mu_t)^2 + (1-\alpha) \times \sigma_t^2}$$

Equation 4-7: run length standard deviation

In equation 4-6 and 4-7, a variable α is defined as the updating rate in [7]. This value affects the speed of the background model updating. In this research, the 0.05 is used as the value of this background updating rate. I_t is block intensity value in the frame t.

4.2.4 Foreground Objects Detection

In this algorithm, a foreground object block is defined as: given a mean intensity value of a particular block, the distance between this value and the mean intensity value of the background model are more than 3 times its standard deviation [41] (Equation 4-8).

$$|x - \mu| > 3\sigma$$

Equation 4-8: a block with mean intensity value x is a foreground block only when this equation is true.

In this HOD algorithm, the foreground object detection starts from the coarsest level and then refines its contours recursively to the highest precision level (Figure 4-6). This hierarchical foreground objects detection algorithm is more efficient than the classical pixel classification algorithms proposed in the stochastic background modelling algorithms in the literature. This advantage is achieved by the inherent nature of the hierarchical structure.

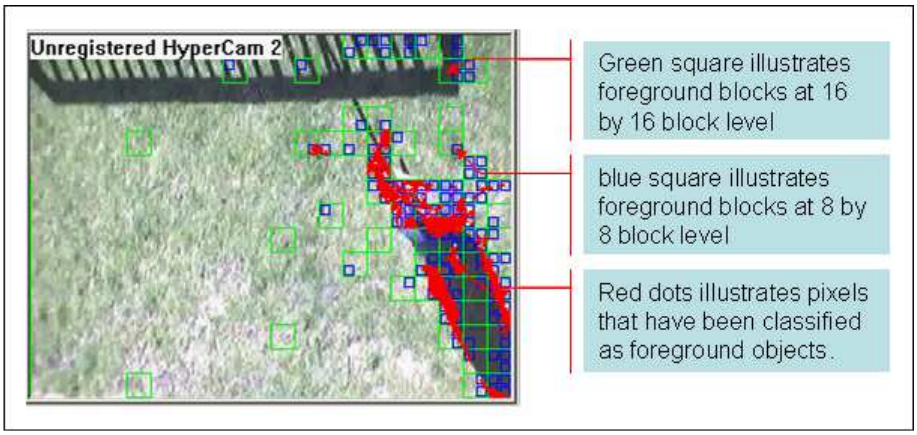


Figure 4-6: A screen image captured from an application that implements the HOD algorithm. The detection operation will only select blocks with their upper blocks already classified as foreground blocks.

4.2.5 Discussion

The HOD algorithm proposed in this section achieves the following advantages by its novel hierarchal structure:

- High performance in foreground object detection.
- Partial background model updating

However, the experiments of an object detection system developed based on this HOD algorithm reveal the following limitations:

1. As the HOD algorithm models background stochastic properties with Gaussian probability distributions, this algorithm does not tolerate non-Gaussian noises.

2. The HOD algorithm is a stochastic based algorithm and therefore it is influenced by the general limitations of such algorithms. For example, the camera in such system is usually required to be perfectly stable. However, this condition cannot be satisfied in this research.

3. In this system, the target objects may stay in the field of view and become part of background. The selective updating scheme of this algorithm rejects these objects to be modelled as background. Therefore, these objects are always segmented as foreground even when they are not objects of interest anymore and the blocks that contain these objects will not be updated in the future.

Based on the experimental research with this HOD algorithm, we conclude that the stochastic based objects detection algorithms are not suitable to the environment we proposed in this research.

4.3 Improved Double Difference Objects Detection Algorithms – Temporal Approaches

In this section, the problems of utilizing the temporal based moving objects detection algorithms are discussed. Firstly, in section 4.3.1, the problems of implementing the double difference algorithm (DDA) proposed in [1] in this research are introduced. In addition to the general DDA algorithm, an algorithm is proposed in section 4.3.2. This novel algorithm, the PEDDA algorithm, is developed based on the general DDA algorithm. Finally, in section 4.3.3, the advantages of the PEDDA algorithm are illustrated.

4.3.1 Implement DDA algorithm in this research

The DDA algorithm proposed in [1] [24] [26] [27] has been discussed in chapter2. In this section, information related to the DDA algorithm is explained. In addition to the implementation details, the problems that we found when utilizing the general DDA algorithm in this research are discussed at the end of this section.

DDA Implementation

As illustrated in the figure 4-7: 3 adjacent frames ((Figure 2-4 [a] [b] [c]) are feed into the DDA processing flow. In the first step, 2 pairs of adjacent frames among 3 images are subtracted from the other (Figure 2-4 [Step 1]). Two adjacent differential images are then generated (Figure 2-4 [d] [e]). In the second step, the 2 adjacent different

images are thresholded by a predefined threshold value to form 2 binary motion masks (Figure 2-4 [f] [g]). Finally, the binary "AND" operation is performed on the two binary motion masks (Figure 4-7 [Step3]). The outcome of the third step is the DDA motion mask image (Figure 2-4 [h]).

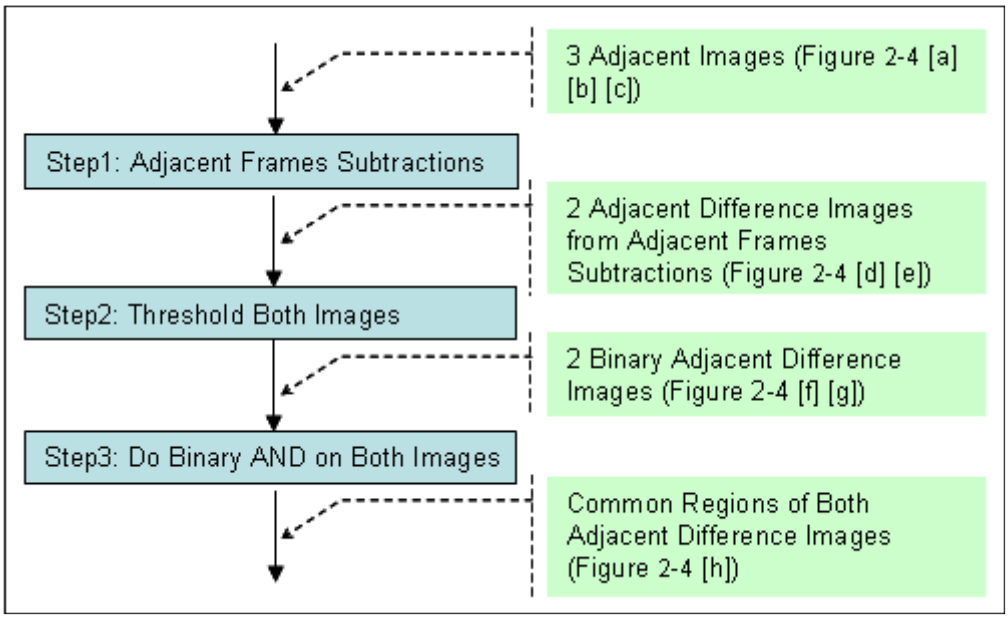


Figure 4-7: The implementation details of the DDA algorithm in this research.

Analysis:

The DDA algorithm overcomes the disadvantages involved in stochastic based approaches and has the following advantages:

1. Although non-Gaussian noises may trigger false positive detection, this error will not be included in the background description, which means that the error influence is limited.

2. Low frequency background movement caused by slow camera movement will not affect the detection results because the mean shift distance involved by such low speed movement are too small to be detected in such temporal based algorithms.

3. Any objects in the field of view will be included as part of the background as soon as they stop moving. This feature is especially useful in this research because only moving objects are of interest. For example, when a ball stops in the field of view, it will be merged into background immediately after it stops.

However, since our research s focused on fast moving objects such as cricket balls, a large, slow moving object, such as the batter in the field of view, will also trigger extra object detection operations. These extra operations not only slow down the candidates' election algorithm but also introduce more false positive errors.

To suppress slow moving objects from being segmented, a modified general DDA algorithm known as the PEDDA algorithm is proposed.

4.3.2 The PEDDA algorithm

The Pixel Exclusion Double Difference Algorithm (PEDDA) algorithm is an improved version of the DDA algorithm. It suppresses low speed movement objects from being segmented by analyzing the 4th image frame in the general DDA algorithm. The PEDDA algorithm is implemented in this research to minimize the influence of low speed moving objects, such as the human body.

Algorithm Design:

For any sequence of 4 image frames, I_{n-2}, I_{n-1}, I_n and I_{n+1}, three adjacent difference frame motion masks, $D_{n-2,n-1}$, $D_{n-1,n}$ and $D_{n,n+1}$ can be calculated by equation 4-9:

$$D_{i-1,i} = Threshold(|I_i - I_{i-1}|)$$

Equation 4-9: adjacent frame difference.

The DDA motion masks, C_{n-1} and C_n can be calculated by equation 4-10.

$$C_i = D_{i-1,i} \cap D_{i,i+1}$$

Equation 4-10: DDA motion mask

In addition to C_{n-1} and C_n, $\overline{C_i}$ is defined as a complementary binary image of the DDA motion mask C_i and is calculated by $\overline{C_i} = Not(C_i)$. Finally, the output of this PEDDA algorithm, the PEDDA motion mask, is given by equation 4-11:

$$P_n = \overline{C_{n-1}} \cap C_n$$

Equation 4-11: The PEDDA motion mask.

Algorithm Implementation:

Figure 4-8 illustrates the implementation details of the PEDDA algorithm in this research.

The first three steps are similar to the DDA algorithm, with the exception that it processes four adjacent images instead of three in as in the general DDA algorithm. By processing four adjacent images, two DDA motion masks are generated. These masks are illustrated in figure 4-9 [a] and [b]. The first DDA motion mask is then inversed to generate an exclusion pixels mask (EPM) (Figure 4-9 [c]). All the pixels that have zero value in the EPM will be excluded from final output. Finally, the second DDA motion mask generated at step 3 is filtered by EPM with a binary AND operation. The PEDDA algorithm's output, the PEDDA motion mask, consists of the remainder pixels in the second DDA motion mask (Figure 4-9 [d]).

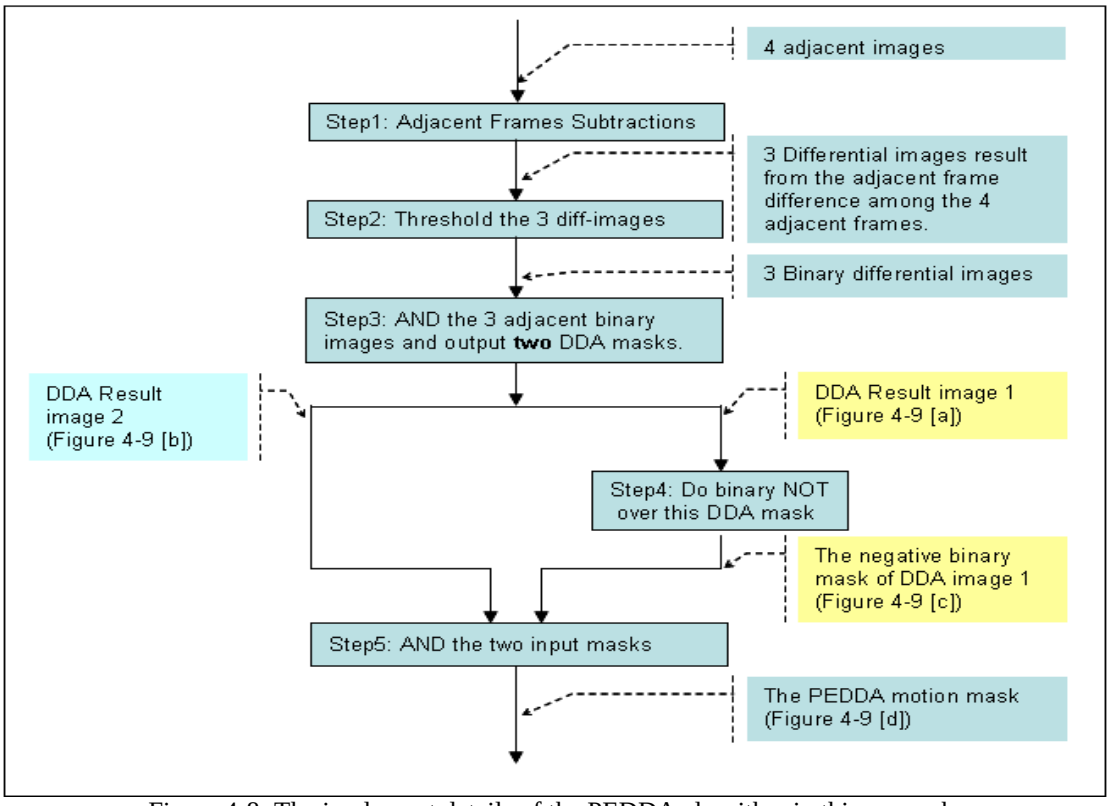


Figure 4-8: The implement details of the PEDDA algorithm in this research.

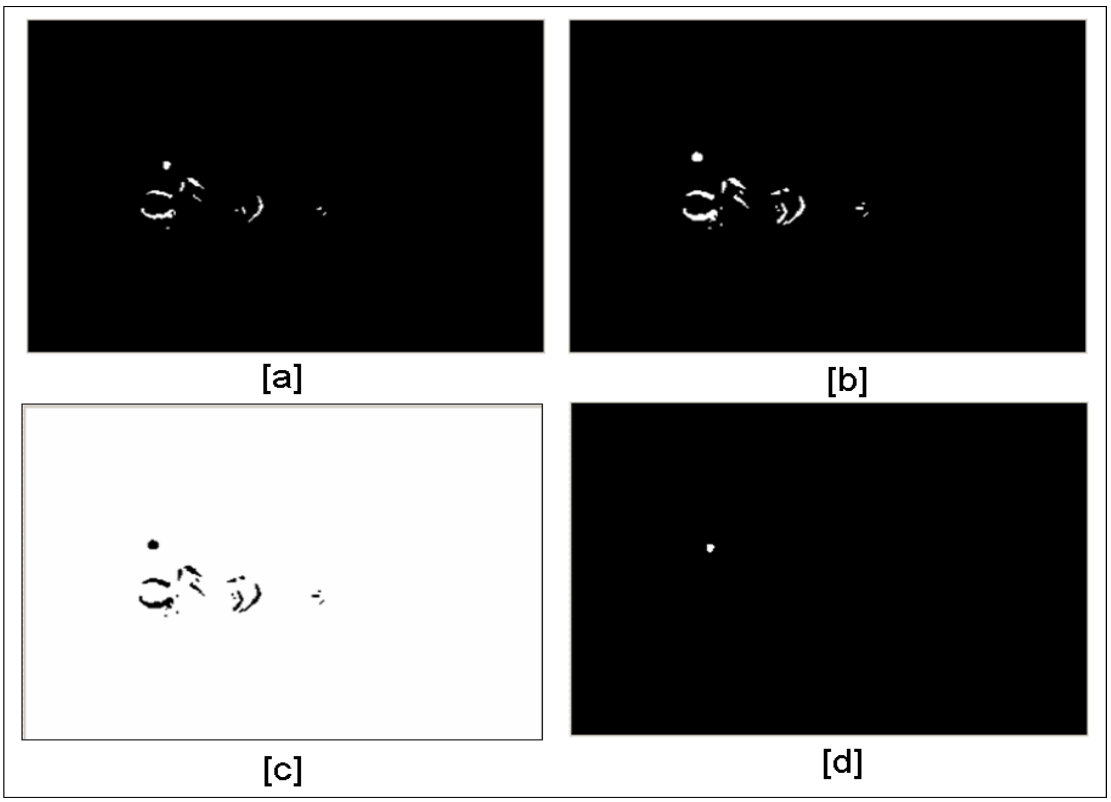


Figure 4-9: Intermediate outputs of the PEDDA algorithm. [a] The first DDA mask. [b] The second DDA mask. [c] The EPM image. [d] The PEDDA result image.

Analysis:

This PEDDA moving objects detection algorithm is an improvement from the general DDA algorithm. In addition to the advantages of general DDA algorithm (list in chapter 2 and section 4.3.1) this novel algorithm reduces the extra object detection involved by low speed non-target objects in the field of view.

In figure 4-10, the first 6 images ([a] to [f]) in the left column are segmented by the general DDA moving objects detection algorithm. In these images, both the cricket ball (a fast moving object that we are interested in) and the batter (a slow moving object that we are not interested in) have been segmented at the same significant level. As a consequence, the target object is merged inside the redundant extra segmented image.

The other 6 images ([g] to [i]) in the right column in figure 4-10 illustrate the image segmentation results of the PEDDA moving objects detection algorithm. Obviously, the slow moving objects, such as the human body, are suppressed and therefore, the fast moving objects that we are interested can be more apparent. The remaining motion trace of slow moving objects can be easily removed by morphological operations, such as erosion, in the steps after PEDDA.

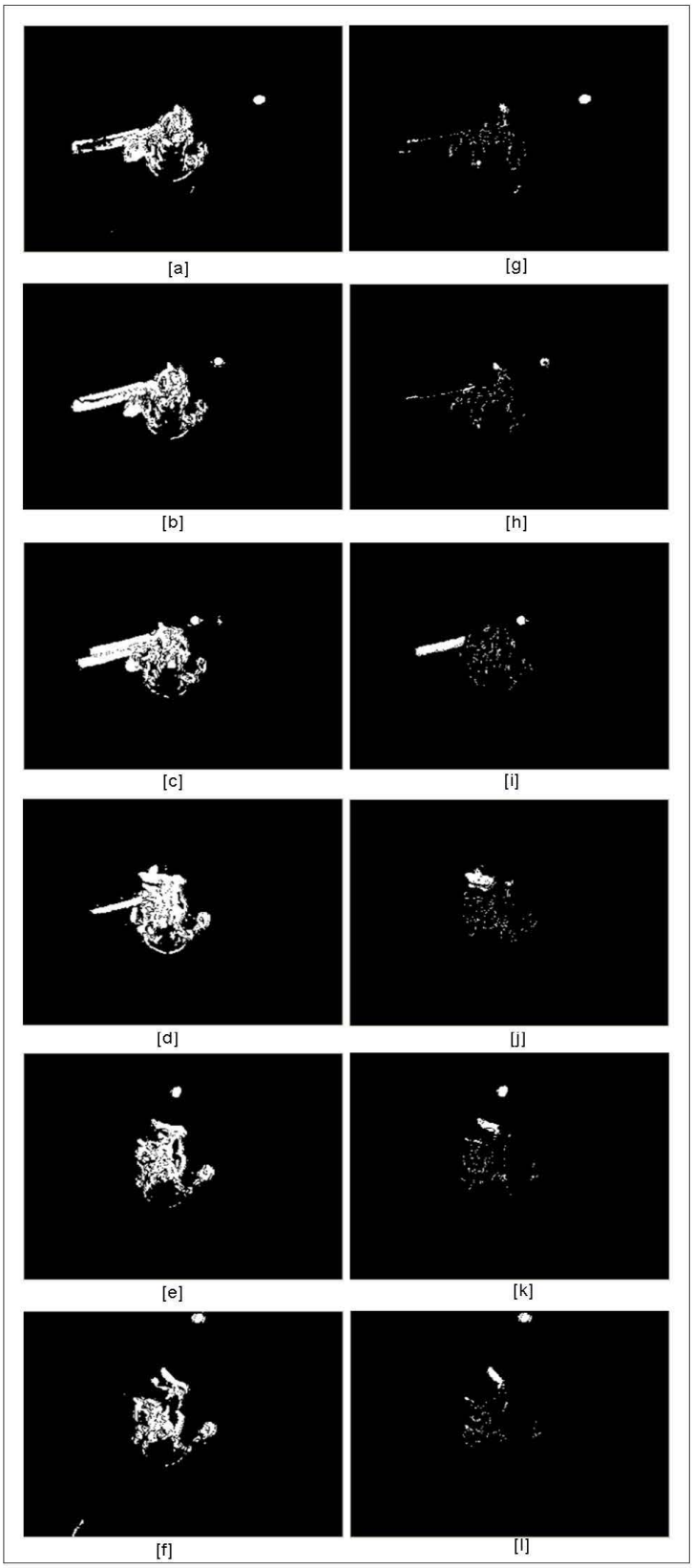


Figure 4-10: Comparison study for the general DDA algorithm and the PEDDA algorithm. [a] – [f]: motion masks generated by the DDA algorithm. [g] – [l]: motion masks generated by the PEDDA algorithm.

4.4 Conclusion

The HOD and PEDDA algorithms were discussed in this chapter. Both were designed in this research to detect fast moving objects. The HOD algorithm has the following advantages:

1. High performance in foreground objects detection.
2. Partial background model updating.

However this algorithm is not suitable for this research because of the intrinsic limitations of stochastic approaches, namely:

1. High image quality requirements.
2. The dilemma of background model updating and foreground objects detection.

On the other hand, it has been shown that the PEDDA algorithm can not only detect moving objects successfully but also suppresses objects of non-interest (slow moving objects). This PEDDA algorithm will be utilized in the experimental tracking system implemented in chapter 6.

Chapter 5

Target Election and Tracking Algorithm

5.1 Introduction

In this chapter, a novel target election and tracking algorithm, the State-based Observation, Analysis and Prediction target election and tracking (SOAPtet) algorithm, is discussed.

The SOAPtet algorithm is a state based algorithm. A deterministic finite state machine (DFSM) is built to constrain the behaviours of the SOAPtet algorithm in order to adapt the algorithm to fit the target's motion properties at different motion stages. This approach will be discussed in the section 5.2.1 in detail.

The SOAPtet algorithm utilizes a trilogy of observation, analysis and prediction to achieve the goals of target election and tracking. A novel score based target election approach is implemented in this trilogy to elect the target that we are interested from all other potential candidates that may be triggered by noise. At the tracking stage, the SOAPtet algorithm uses a Kalman filter to predict the trajectory features of a target. An error tolerance parameter is designed to compensate for the prediction errors. The value of tolerance parameter can be adjusted automatically by the SOAPtet algorithm

during the observation step of tracking states. A detailed discussion shown of the Observe, Analysis and Prediction steps is provided in section 5.2.2.

5.2 The State-based "Observation, Analysis and Prediction" Target Election and Tracking Algorithm

A deterministic finite state machine (DFSM) is implemented in the SOAPtet algorithm. This DFSM determines the behaviours of the SOAPtet algorithm. In this section, this DFSM is discussed in detail including the 5-tuple definition and the state transition diagram.

5.2.1 The Deterministic Finite State Machine

5.2.1.1 The 5-tuple definition

Mathematically, a DFSM is defined by a 5-tuple [28] [29]:

$M = \{S, \Sigma, \delta, S_0, A\}$, where

- S: A finite set of states.
- Σ: A finite set of input symbols (also called the *input alphabet*).
- δ: $S \times \Sigma \to S$ is the finite transition function.
- S_0: The initial state (also called start state).
- A: $A \subseteq S$ is a finite set of accept states (also called accepting states or final states).

The DFSM implemented in the SOAPtet algorithm using the 5-tuple definition is as follows:

1. The finite state S:

   In the context of the SOAPtet algorithm, 6 states are defined to describe and constrain the behaviours of the algorithm itself. These states are:

   1) The “inward targets detection” state.
   2) The “inward target tracking” state.
   3) The “outward targets detection” state.
   4) The “outward target tracking” state.
   5) The “idle” state.
   6) The “stop” state.

   The 6 states are summarized in the Table 5-1 and will be further discussed in the rest of this chapter.

States Name	**State ID**	**Initial**	**Accept**	**Description**
Inward targets detection	S1	YES		The algorithm detects targets moving inward.
Inward targets tracking	S2			The algorithm tracks targets moving inward.
Outward targets detection	S3			The algorithm detects targets moving outward.
Outward targets tracking	S4			The algorithm tracks targets moving outward.
Idle	S5		YES	The algorithm is paused for a predefined period.
Stop	S6			The algorithm is stopped.

Table 5-1: The 6 states of DFSM in the SOAPtet algorithm

2. The finite set of input symbols $\sum$:

The SOAPtet algorithm has 11 input symbols in its inbuilt DFSM. These input symbols are:

1) [a]: will be input in the DFSM when an inward candidate is detected.
2) [b]: will be input in the DFSM when an inward target is elected.
3) [c]: will be input in the DFSM when the inward target is under tracking.
4) [d]: will be input in the DFSM when the inward target merges into the batter's influence area.
5) [e]: will be input in the DFSM when the target (either inward or outward) is close to the image boundaries.
6) [f]: will be input in the DFSM when an outward candidate is detected.
7) [g]: will be input in the DFSM when an outward target is elected.
8) [h]: will be input in the DFSM when timer runs out.
9) [i]: will be input in the DFSM when the target disappears.
10) [j]: will be input in the DFSM when the SOAPtet algorithm is stopped.
11) [k]: will be input in the DFSM when the SOAPtet algorithm is restart.

These 11 input symbols are summarized in the following table:

No.	Symbol	Comment
1	[a]	Inward candidate is detected
2	[b]	Inward target is elected
3	[c]	Inward target is tracked
4	[d]	Inward target merges into batter's influence area
5	[e]	The target is close to image boundaries
6	[f]	Outward candidate is detected
7	[g]	Outward target is elected
8	[h]	time out
9	[i]	The target disappears
10	[j]	The SOAP algorithm is stopped
11	[k]	The SOAP algorithm is restarted

Table 5-2: The 11 input symbols in the DFSM built in the SOAPtet algorithm.

3. a finite set of transition function δ:

   The transition function is presented in table (Table 5-3).

	[a]	[b]	[c]	[d]	[e]	[f]	[g]	[h]	[i]	[j]	[k]
S1	**S1**	**S2**								**S6**	
S2			**S2**	**S3**	**S5**			**S1**	**S1**	**S6**	
S3		**S4**				**S3**		**S1**		**S6**	
S4					**S5**		**S4**	**S1**	**S1**	**S6**	
S5								**S1**		**S6**	
S6											**S1**

Table 5-3: Transition function in DFSM.

4. The start state S_0:

   S_0 is the first state of the DFSM. The initial state S_0 is defined as the "Inward targets detection" state (S1).

5. The finite set of accept states A:

   In the SOAPtet algorithm, we define the "Idle" state as the accept state because when the SOAPtet algorithm arrives at this state, the trajectories of a target can be extracted (accepted) using the recorded beacons.

The built-in DFSM of the SOAPtet algorithm has now been defined by using the 5-tuple introduced in [28] [29]. The operation of this DFSM will be explained by a state transition diagram in the next section.

5.2.1.2 The State Transition Diagram

The state transition diagram is a graphical representation of states and transitions in the DFSM [29]. It depicts the relationships between the system states and the input symbols that cause the system to change from one state to another [30].

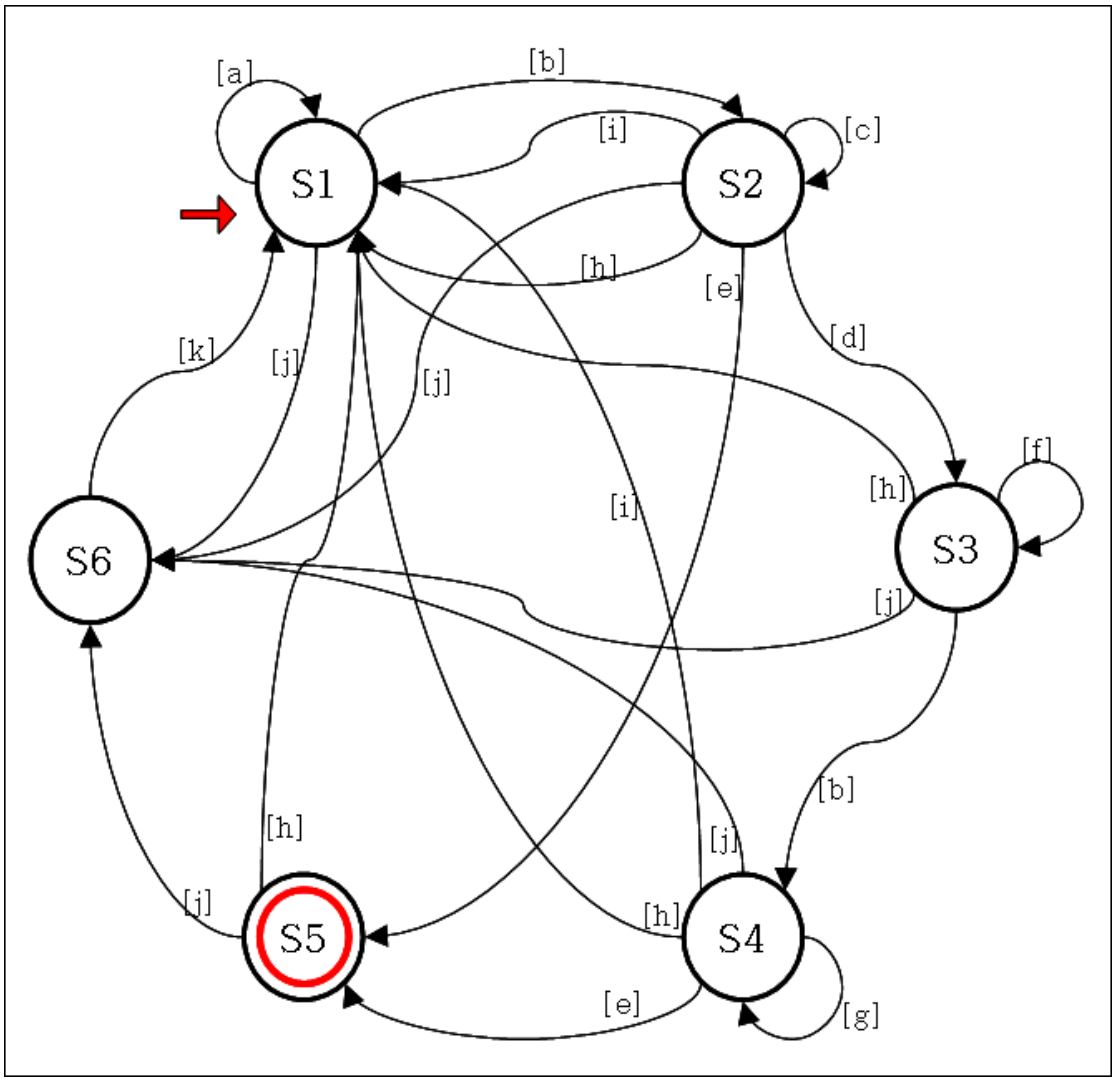


Figure 5-1: The state transition diagram of the DFSM implemented in the SOAPtet algorithm.

Figure 5-1 illustrates the state transition diagram of the DFSM that is implemented in the SOAPtet algorithm. In this diagram, the states are depicted by circles. The initial

state, S1, is marked by a thick arrow and the accept state, S5, is indicated by double circles. The arrows between states indicate state the transition between two states depending on the input (marked in square brackets).

As shown in the figure 5-1, this DFSM starts from state 1: "**Inward targets detection**". In this state, the SOAPtet algorithm observes all inward candidates. The algorithm assigns a score to each of the observed candidates (this will be discussed in detail in section 5.2.2). This score indicates the quantized likelihood of a candidate being a target. In this state:

- If there is no candidate that has a score higher than the predefined threshold value, the SOAPtet algorithm will put symbol [a] into the DFSM; the DFSM will remain at the initial state.
- If a candidate has been elected because its score is greater than the predefined threshold value, the symbol [b] is inserted into the DFSM, and the current state will change to the "Inward target tracking" state.
- If the stop event is input into the DFSM, the current state will change to the "Stop" state (S6).

In the "**Inward target tracking**" state, the SOAPtet algorithm tracks the elected target from frame-to-frame. One of the 6 events happens in each frame when the DFSM is in this state:

- If the target object is detected and its position is neither close to the batter nor to any of the image boundaries, the symbol [c] will be input into this DFSM. The DFSM will remain at S2 in this case.
- If the target object is detected and its position is close to the batter's influence area (the distance between them is less than a predefined threshold value), then the event [d] is inserted insert into the DFSM and the current state will switch to state S3, "Outward targets detection".
- If the target object is detected and its position is close to any of the image boundaries (the distance between them is less than a predefined threshold value), then event [e] is put into the DFSM and the current state will change to S5, "Idle state". The "Idle state" is an accept state, which means that if the DFSM reaches this state, the target trajectories will be accepted and recorded into the database for future use.
- If a time out event [h] is insert into the DFSM, its state will be reset to S1. The time out event is generated by a timer that initiates when the current state changes either from S1 to S2 or from S3 to S4. This timer is one of the state recovery measurements designed for the SOAPtet algorithm to survive in very noisy image conditions.
- If the target disappears even after readjust threshold and prediction error estimation (this will be discussed in section 5.2.2) the symbol [i] is input into the DFSM and the DFSM will reset itself to the initial state.
- Finally, if the stop event is input into the DFSM, the current state will change to "Stop" state (S6).

In the "**Outward target detection**" state, the SOAPtet algorithm observes all outward candidates. The algorithm assigns a score, which will be discussed in detail in section 5.2.2, to each of the observed candidates. This score indicates the quantized likelihood of this candidate being a target. In this state:

- If there is no candidate that has a score higher than the predefined threshold value, the SOAPtet algorithm inputs symbol [f] into the DFSM; the DFSM will remain in state S3.
- If a candidate has been elected because its score is greater than the predefined threshold value, the symbol [b] is insert into the FDSM, and the current state will change to the "Outward target tracking" state.
- If a time out event [h] is inserted into the DFSM at this state, the DFSM will be reset to S1.
- Finally, if the stop event is input into the DFSM, the current state will change to the "Stop" state (S6).

In the "**Outward target tracking**" state, the SOAPtet algorithm tracks the elected target from-frame-to-frame. One of the 5 events occurs in each frame when the DFSM is in this state:

- If the target object is detected and its position is not close to any of the image boundaries, the symbol [g] is input into this DFSM. The DFSM will remain in S4.
- If the target object is detected and its position is close to any of the image boundaries (the distance between them is less than a predefined threshold

value), then the event [e] is input into the DFSM and the current state will change to S5, “Idle state”. The target trajectories will be accepted and recorded into the database for future use because the S5 is an accept state.

- If a time out event [h] is inserted into the DFSM, its state will reset to S1. The time out event is generated by a timer that initiates when the current state changes either from S1 to S2 or from S3 to S4.
- If the target disappears even after readjusting the threshold and prediction error estimation (which will be discussed in section 5.2.2) the symbol [i] will be input into the DFSM and the DFSM will reset to the initial state.
- Finally, if the stop event is input into the DFSM, the current state will change to the “Stop” state (S6).

In the “**Idle**” state, the SOAPtet algorithm pauses for a short period to allow the computation resource (such as CPU time) to be used for other tasks. This feature is especially useful when the algorithm is implemented in a system that requires heavy graphical drawing tasks. Furthermore, as this state is an “accept state”, if the DFSM arrives at this state, the target trajectory information will be stored into database for future analysis. In this state:

- If the algorithm stops, the stop event is input and the current state will change to S6.
- If the time out event [h] is put, the current state will reset to the initial state.

In the “**Stop**” state, the SOAPtet algorithm is halted until a restart event, [k], occurs.

In this section, we introduced a deterministic finite state machine, which establishes the core framework of the SOAPtet target election and tracking algorithm, using the 5-tuple definition and its state transition diagram. In the following section, the effect of this DFMS to the SOAPtet algorithm will be explained in detail when we introduce other highlights of this algorithm.

5.2.2 The Trilogy of Observation, Analysis and Prediction

The DFSM discussed in the last section forms the core framework of the SOAPtet target election and tracking algorithm. A trilogy of Observation, Analysis and Prediction is implemented in the target election and tracking states of the DFSM framework to achieve the goal of:

- Electing the target from its candidates
- Tracking of target when it is observable

In this section, the “Observation”, “Analysis” and “Prediction” steps will be introduced in detail to illustrate the way that this SOAPtet algorithm achieves the two goals mentioned above.

5.2.2.1 Concept Definition

Firstly, before we discusses the “Observation”, “Analysis” and “Prediction” steps, several concepts and jargons that will be used in the discussion are introduced:

- Batter’s influence area:

  The batter’s influence area is a round area in the centre of an image that contains the batter (Figure 5-2). This area is defined to isolate the image that contains the batter from the rest of images because the accuracy of tracking a cricket ball in this area is reduced by occlusion. This area is also an indicator for the target trajectory variation, as the probability of the target being hit by the bat is very high if the target flies into this area.

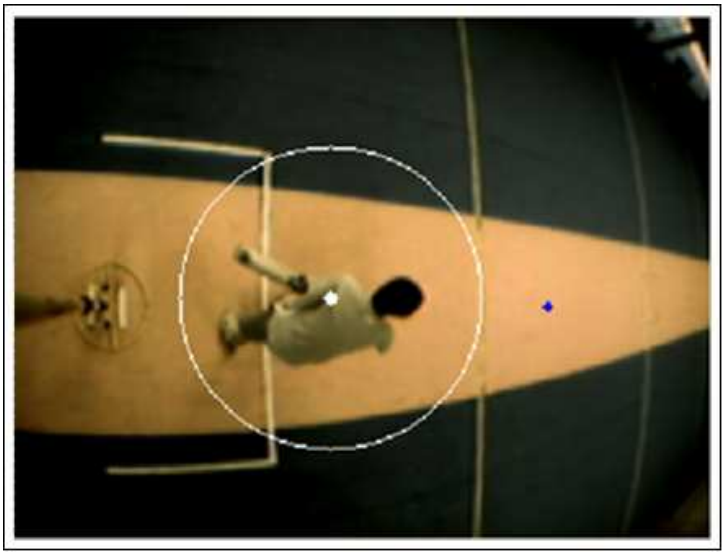


Figure 5-2: The batter’s influence area.

- Inward-detection-area:

The inward-detection-area is an area in the image where any target moving in the inward direction will pass through. In this research, as a cricket ball tracking system is used to implement the proposed algorithms, the inward-detection-area is defined as figure 5-3 [a]. Due to the rules of this game, every ball that flies towards the inward direction is expected to pass through the rectangular area marked in the figure 5-3 [b]. The inward target detection process is carried in this area.

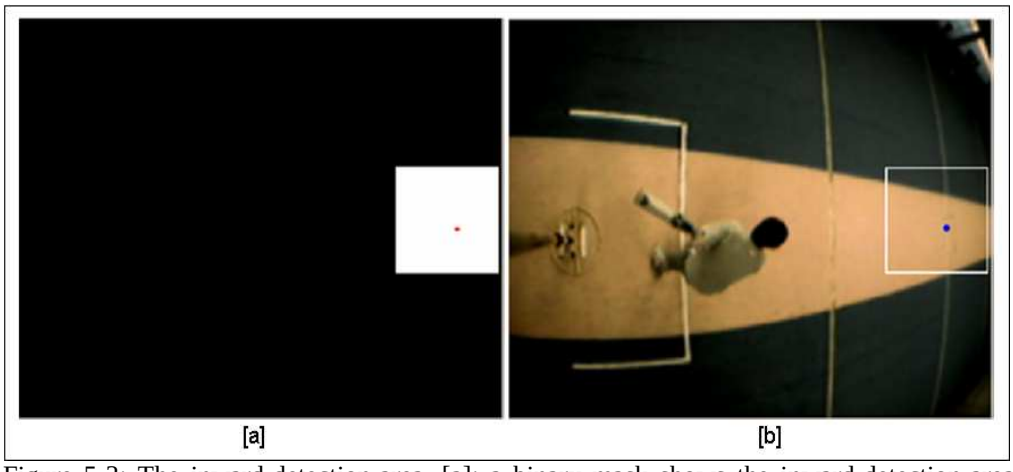


Figure 5-3: The inward-detection-area. [a]: a binary mask shows the inward-detection-area (the white rectangular area); a red dot shows several pixels around are segmented as moving object is detected[b]: the location of the inward-detection-area in the cricket game field; the blue dot shows the position of an inward cricket ball.

- Outward-detection-area:

The outward-detection-area is an annular area surrounding the batter's area of influence (Figure 5-4). As the target's trajectory has high probably to be changed by the influence of batter, the historic trajectory information of an inward target is useless to predict the appearance of its outward trajectory. The outward target detection process has to be carried out over candidates that appear in the outward-detection-area.

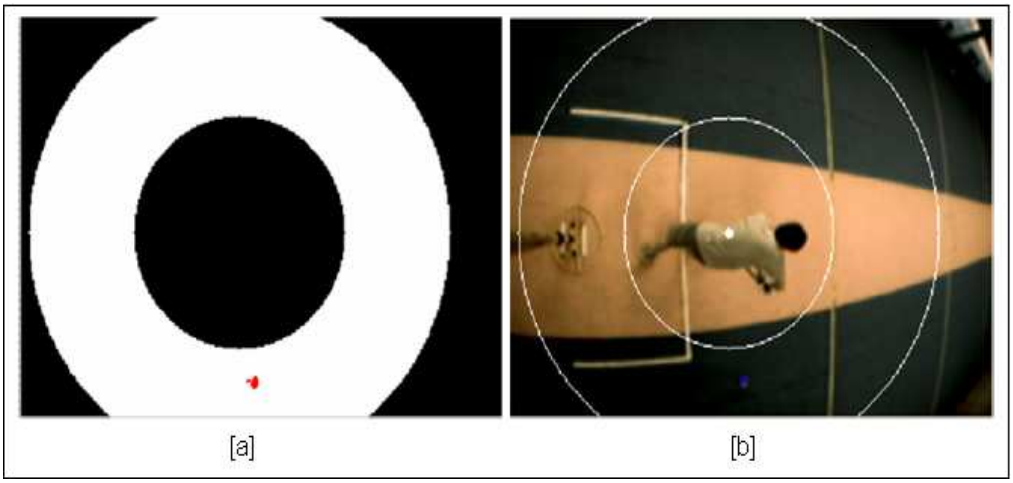


Figure 5-4: The outward-detection-area. [a] a binary mask shows the outward-detection-area (the white annular area); a red dot shows several pixels around are segmented as moving object is detected. [b] The location of the outward-detection-area. The blue dot in the image shows the position of an outward cricket ball.

- Candidate:

In the target detection states, such as S1 and S3 states, the algorithm cannot distinguish which "target" is triggered by the moving target of interested and which is triggered by noise. These segmented image blobs are named as

"candidates". A candidate is the location where a moving object or noise is detected.

- Prediction area:

In this algorithm, a prediction of the appearance location of a target in the next frame is calculated by a Kalman filter [31] using the trajectory history. The estimated position where the target may appear in the next frame and a small area surrounding this position is defined as the prediction area of a target (Figure 5-5).

Figure 5-5: The prediction area.

- Candidate buffer:

  The candidate buffer is an array that contains all the candidates that have been detected by an image segmentation algorithm. Each candidate in the buffer is represented by its centre point's coordination. This is a temporal buffer and is refreshed once a new frame arrives.

- Prediction information buffer (PID):

  The PID is a data structure implemented to store prediction information between frames. The prediction information is calculated by the target's trajectory properties that have been observed in the past frames and will be verified in the next frame.

  A PID is consists of:

  - A pair of values that indicate the coordinates of the centre of a prediction area.
  - An integer value that indicates an estimation of the prediction error.
  - A floating point number that indicates the threshold of an image segmentation algorithm that should be used to segment the image area with this prediction.
  - A set of coordinates indicating candidates that have been observed in this prediction area.

- Trajectory Information Buffer (TIB):

  A candidate's TIB is a data structure implemented in the SOAPtet algorithm to store the trajectory history of a particular target. In the case where more than one candidate is present when the DFSM is at detection state, a set of candidates buffers are used and each chain contains each candidate's trajectory history. The prediction of a particular candidate is done by analysing the trajectory history stored in this buffer. When a candidate has been classified as noise, the buffer that contains this particular candidate will be purged.

  A TIB is consists of:

  - An array that contains each beacon's position along the trajectory of a candidate.
  - An integer score value that indicates the likelihood of the candidate contained in this buffer is the target object. The usage of this value will be discussed later.

For the rest of this section, the "Observation", "Analysis" and "Prediction" steps in the SOAPtet algorithm will be discussed in detail.

5.2.2.2 The Observation Step

As the SOAPtet algorithm is a state based algorithm, the tasks that have been achieved in the "Observation" step is different based on the state of the algorithm. Therefore, the observation step is discussed with the state conditions as follows:

When the algorithm is in detection state (S1 and S3), the observation step completes the following tasks:

- Task 1: Classify the observed candidates into two groups: (1) the candidates that appear within prediction areas which are inferred by the candidates information in the last frame; (2) the candidates that did not appear in any prediction areas.
- Task 2: Record the positions of the candidates that appear in the inward-detection-area (in S1 state) or the outward-detection-area (in S3 state) into the TIB. If the candidate is classified as one beacon along a trajectory, then the position of this candidate is stored in the existing trajectory buffer. Otherwise, a new trajectory buffer is allocated for this candidate.
- Task 3: Purge an information candidate buffer if no candidate is found in the prediction area calculated by the trajectory history stored in this buffer.

Figure 5-6 illustrates the flowchart diagram of the "Observation" step in detection states (S1 or S3). A candidate buffer is put into the "Observation" step, which stores all the candidates that have been detected in the current frame. Then, each prediction area in the PID is tested by all the candidates. If one candidate is found in a prediction

area, this candidate is moved to the end of the TIB that related to this prediction area. In the case where more than one candidate has been detected in one prediction area, the average position of these candidates is used. Finally, a new TIB will be created for the remaining candidates, which have appeared in the image for the first time.

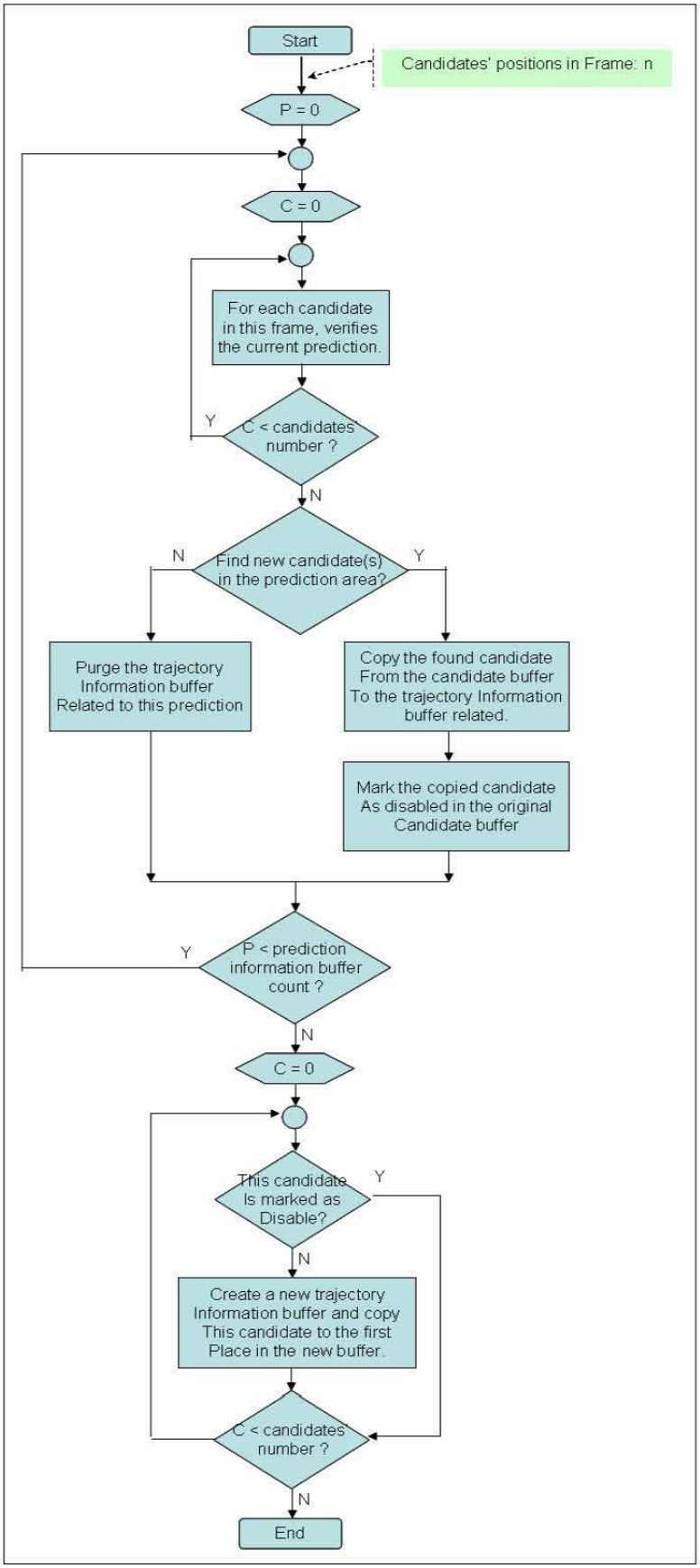


Figure 5-6: The flowchart of the "Observation" step in both detection states (S1 and S3).

In the target tracking states (S2 and S4), the observation step completes the following tasks:

- Task 1: Find the target in the predication area. If the target is found, copy the new detected position to this target's TIB.
- Task 2: If the target cannot be found in the current prediction area, the observation step will increase the prediction area diameter and decrease the threshold values used in the system automatically to compensate for prediction error and background texture variation. Then it will notify the algorithm to redo the object detection and image segmentation process with a new threshold value and find the target in the increased prediction area.
- Task 3: If the target cannot be found, even after the automatic adjustment of prediction area size and threshold, the disappearance event is reported to the algorithm by releasing a flag.

Figure 5-7 illustrates the flowchart diagram of the "Observation" step in tracking states S2 or S4. A candidate buffer is put into the "Observation" step, which stores all the candidates that have been detected in the current frame. Then, each candidate in this buffer is verified by the prediction area that is calculated by the tracking target's trajectory properties recorded in the past. If one candidate is found in the prediction area, its position will be appended at the end of the target's TIB. If two or more candidates are found in the prediction area, their average position will be used as the position of the target and be appended at the end of the target's TIB as well. On the other hand, if the "Observation" step cannot find any candidate in the prediction area, it will first adjust the radius of the prediction area to tolerate more prediction error. If

the radius of the prediction area has exceeded a predefined maximum value, the algorithm will then adjust the threshold of the image segmentation algorithm and notify the system to redo moving object detection and image segmentation process. Finally, if the target cannot be found even when the lowest acceptable threshold value was used, the “Observation” step will report a “missed tracking” event to the algorithm. The tracking state will be analysed in the next step: “Analysis”

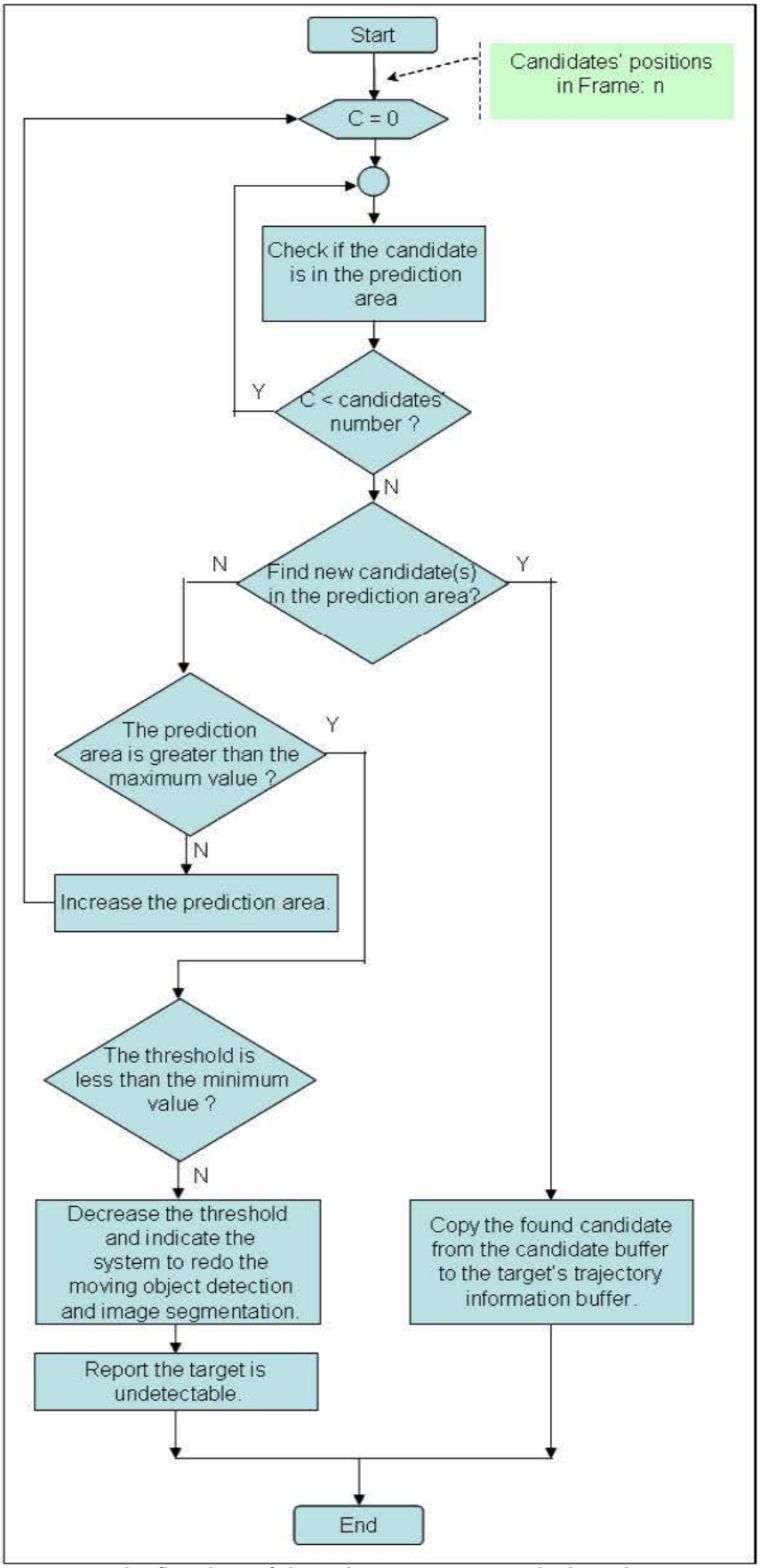


Figure 5-7: The flowchart of the "Observation" step in both tracking states S2 and S4.

5.2.2.3 The Analysis Step

The “Analysis” step is designed to analyse the trajectories information that the algorithm learnt from the “Observation” stage. It has two main goals:

- Assign a score to each observed candidates and elect the target out of the candidates based on the score in the target detection states (S1 and S3).
- Monitor the trajectory of the target in the tracking states (S2 and S4). Report to the system when it believes the target is close to the batter’s influence area (S2) and/or the image boundaries (S2 and S4).

When the SOAPtet algorithm is in target detection states (S1 and S3), the candidates’ trajectory information are stored in the TIB in the “Observation” step of each frame. These trajectory data are analysed by the algorithm to elect the target of interest (in this project the cricket ball) from all the trajectories. Figure 5-8 [a] illustrates the image segmented results and the remaining energy (the gray pixels in the white rectangular) caused by noise. In the figure 5-8 [b], the dark gray dots are a visualization of the TIBs the algorithm created in the “Observation” step of this frame.

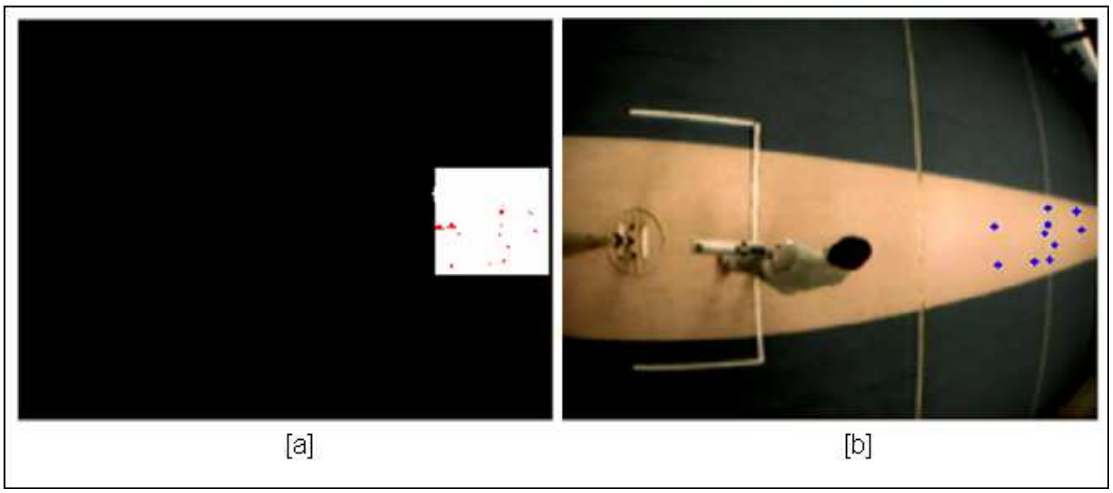


Figure 5-8: The candidates caused by noise in the S1 state.

A novel score based approach is designed to elect the target interest its candidates. This approach is implemented in the “Analysis” step when the SOAP algorithm is in the target detection states (S1 and S3).

Each candidate that has been detected for the first time is assigned a base score, for example ten. As a new candidate will be stored in a new TIB, the buffer gets a score of ten because of the new object. (Figure 5-9)

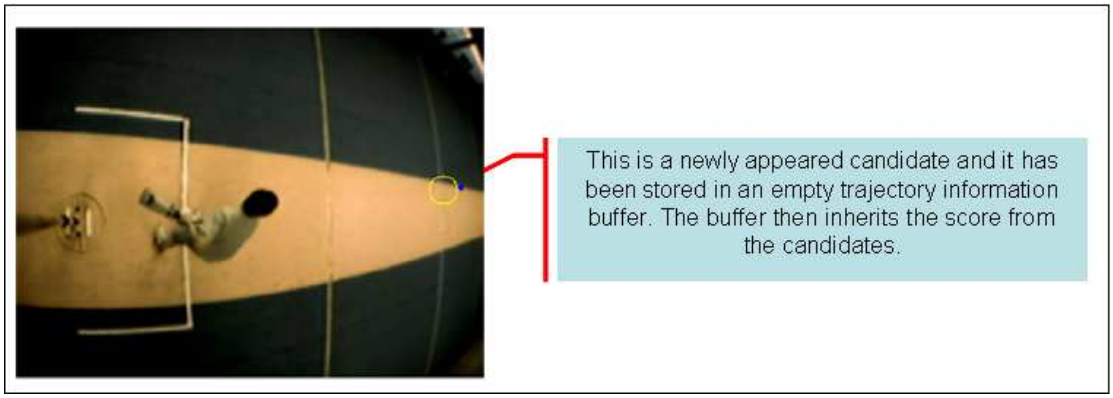


Figure 5-9: A score is assigned to a new candidate and the trajectory buffer.

Each candidate has a prediction area that indicates a small part of the image area where if another candidate is found in this area in the next frame, then there is a very high probability to believe that the two candidates are caused by the movement of the same object. As shown in figure 5-10, a candidate is found in the prediction area of the last frame (indicated by the circle close to the right image boundary). The two candidates appeared in two adjacent frames are linked together by the algorithm when the properties of the second candidate are appended to the TIB where the first one is stored. Therefore, this particular trajectory, represented by the TIB, gets another ten marks for having a second member.

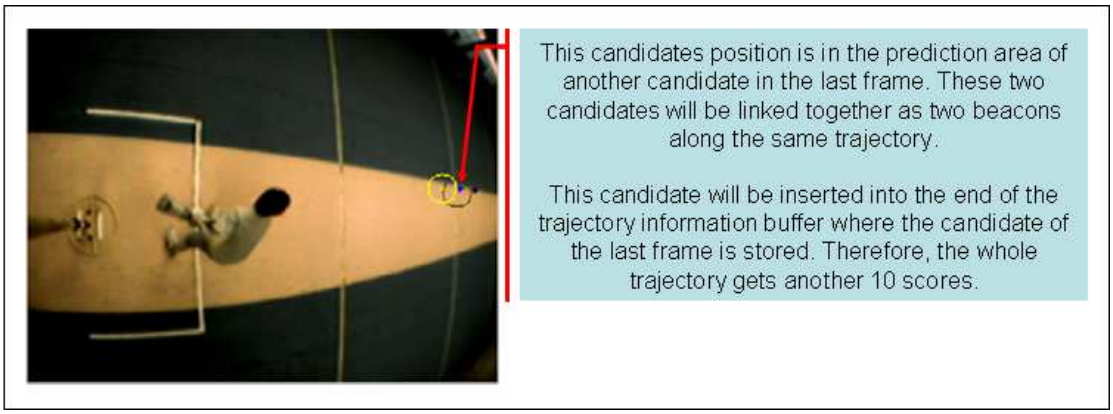


Figure 5-10: A new candidate contributes to the score of the whole chain.

In order to compensate for the noise problem, such as figure 5-8, an image evaluation approach is developed in the ‘Analysis” step in target detection states (S1 and S3). An image frame is classified into three levels based on the number of candidates that have been found in the detection area. If only one candidate is found in the detection area, this image is marked as a good frame. In the good frame, a candidate can get a five marks bonus in addition to the ten marks it has already received. If more than one candidate has been found in the detection area, but the number of candidates is less than or equal to five, the image is evaluated as a “normal” quality image. In this case,

no bonus marks are awarded but the candidate can still get ten marks. In the last case, if more than five candidates have been found in the detection area, then eight marks will be subtracted from the candidate score because in this case, the reliability of this candidate is reduced. In another words, it is more likely that the candidate is affected by noise instead of the target. The threshold values used to classify the image quality were obtained though experimental study using an implementation of this algorithm.

Through this scoring system, the number of frames that a target should be observed before it is elected can be automatically adjusted. For example, in our implementation, a score of 29 score (gained using experimental study) is used as the threshold value to elect the target out of all the candidates. In another words, the first trajectory that has earned more than 29 marks in total (the sum of all candidates' marks along the trajectory) will be granted as the tracking target we are looking for. In the "good" image condition, a consecutive two frames of appearance are enough to allow the algorithm to elect this trajectory as the tracking target. On the other hand, in the worst situation, even 14 consecutive appearances are not sufficient to convince the algorithm to believe that the appeared objects are caused by a moving target.

Figure 5-11 illustrates the "Analysis" step of the SOAPtet algorithm during the target detection states (S1 and S3).

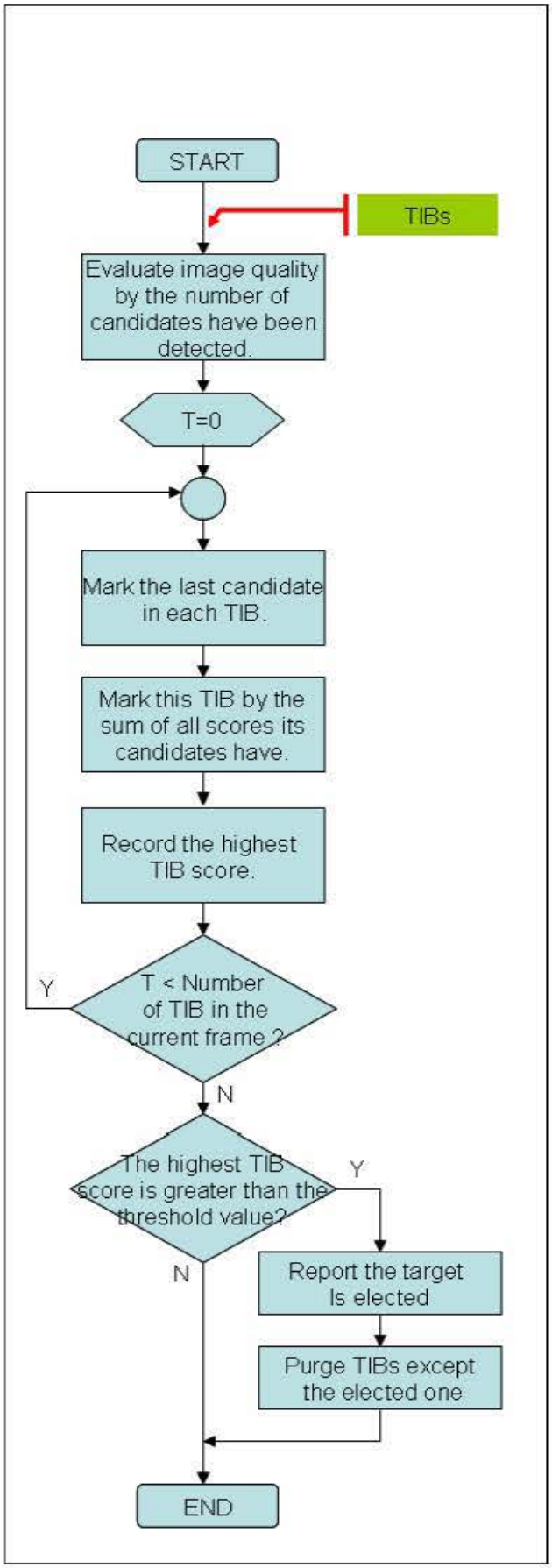


Figure 5-11: The flowchart of the “Analysis” step in both detection states S1 and S3.

When the SOAPtet algorithm is in the inward target tracking states (S2), the "Analysis" step analyses the latest beacon's position in the TIB. When the distance between this beacon and the batter's influence is less than a pre-defined threshold, the "Analysis" step will report that the target is merged into the batter's influence area. This event will be inserted in to the DFSM to drive the current state from S2 to S3. Furthermore, if the position of the latest beacon of a trajectory is close to the image boundary, then another report will be sent to the algorithm to indicate that the target is going to fly out of the field of view.

In the outward target tracking state (S4), only the boundary checking is carried out.

Figure 5-12 illustrates the flowchart diagram of the "Analysis" step of the target tracking state in the SOAPtet algorithm.

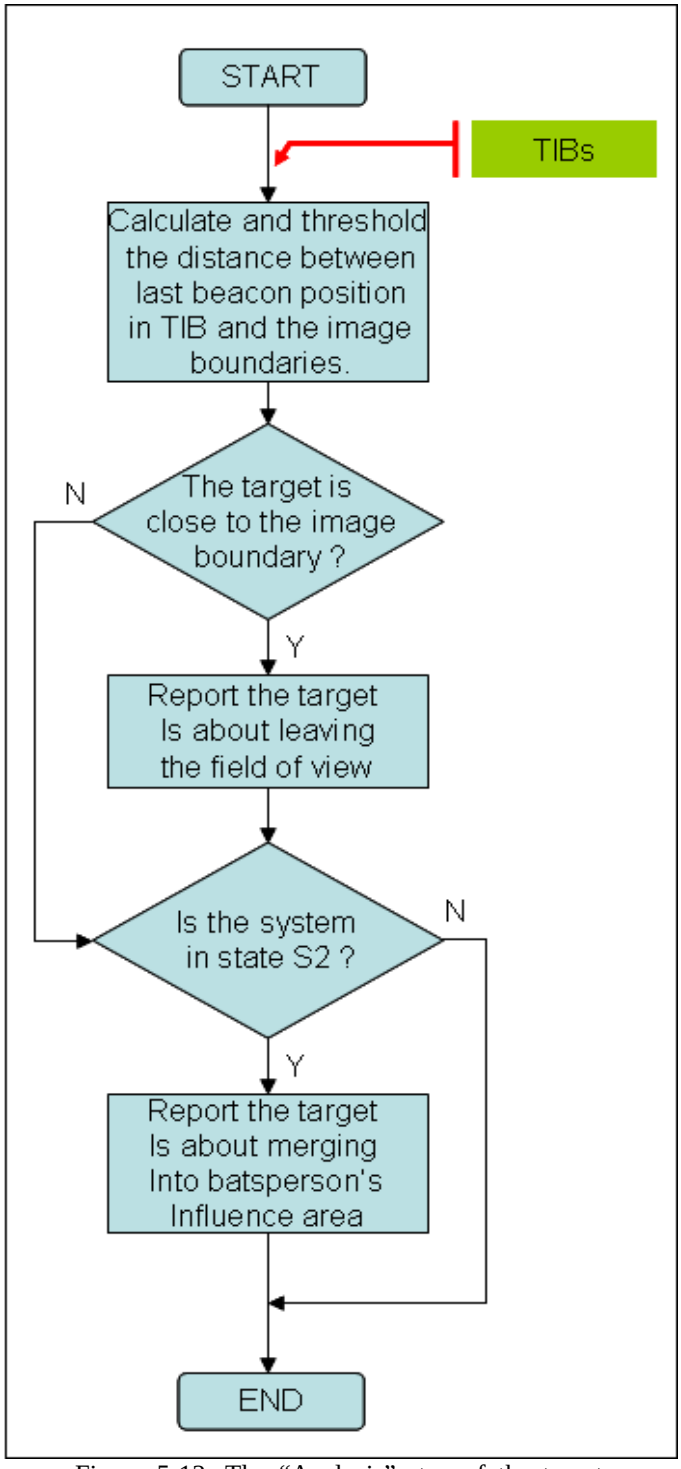


Figure 5-12: The "Analysis" step of the target tracking states S2 and S4.

5.2.2.4 The Prediction Step

The "Prediction" step in the SOAPtet algorithm is designed to predict the position that the tracking target will appear in the next frame based on its known trajectory features. A Kalman filter [31] [32] [33] based approach is implemented in the prediction step of the SOAPtet algorithm. In the following part of this section, the prediction approach is introduced in three aspects: the motion model, the initial state and the error tolerance approach.

1) The mathematical representation of the motion model

The prediction approach in this SOAPtet algorithm can be mathematically described as a state vector transformation in equation 5-1:

$$\begin{bmatrix} X_{k+1} \\ Y_{k+1} \\ \Delta X_{k+1} \\ \Delta Y_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_k \\ Y_k \\ \Delta X_k \\ \Delta Y_k \end{bmatrix}$$

Equation 5-1: The mathematical description of the prediction approach used.

In the equation 5-1:

- (X_k, Y_k) gives the location of the beacon detected in the current frame.

- (X_{k+1}, Y_{k+1}) represents the location of the target to appear in the next frame.
- ΔX_k and ΔY_k represent the displacement between the beacon in the current frame and the one observed in the last frame.
- ΔX_{k+1} and ΔY_{k+1} represent the displacement between the position of the beacon to be observed in the next frame and the position of the beacon observed in the current frame.

2) The initial states

As the displacement is used in the prediction, an initial value of the displacement distance for the first beacon of a trajectory is required to be assigned. In this algorithm, a distance value *d* is predicted as the default initial displacement distance. The initial placement of both *x* and *y* directions are calculated by placement distance d and, position of the first beacon and the algorithm state.

The figure 5-13 shows the initial displacement in state S1, the inward target detection stage. In this state, the candidate (X_1, Y_1) should move towards the batter who is in the centre of the image (X_{bat}, Y_{bat}). The angle θ can be calculated based on this assumption by equation 5-2 and therefore, the displacement at *x* and *y* direction can then be calculated by equation 5-3 and equation 5-4 respectively.

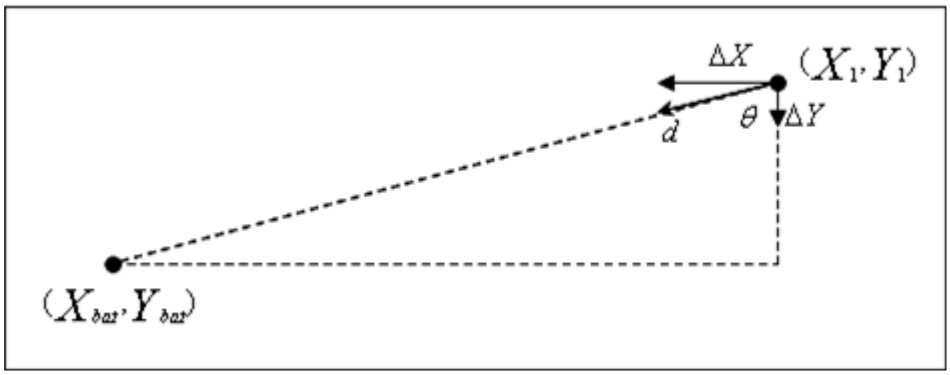


Figure 5-13: The initial displacement in state S1.

$$\theta = \arcsin[\frac{(X_{bat} - X_1)}{(\sqrt{(X_{bat} - X_1)^2 + (Y_{bat} - Y_1)^2})}]$$

Equation 5-2: the θ value.

$$\Delta X = d \times \sin(\theta)$$

Equation 5-3: the displacement in *x* direction

$$\Delta Y = d \times \cos(\theta)$$

Equation 5-4: the displacement in *y* direction

In the outward target detection state S3, a similar approach is used. Figure 5-14 illustrates the outward target case. The equations used in this state are similar to the ones used in the S1 state.

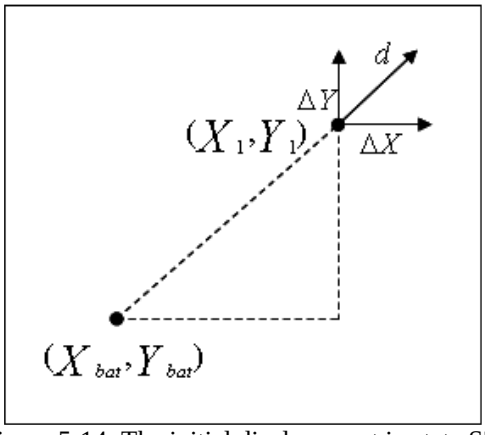


Figure 5-14: The initial displacement in state S3.

3) The prediction error compensation approach

Prediction errors may occur in the system because of inaccurate *posteriori* knowledge and an incorrect motion model. As mentioned in the "Observation" step, if multiple candidates have been found in one prediction area, the mean position of these candidates will be used as the position of the target. This approach may introduce observation error [31] into the system. Furthermore, a succinct motion model is used in this system for computational efficiency reasons. In this model, the target is modelled by a linear trajectory. However, in this context, the trajectory of a cricket ball can be very complex. In this case, to predict a target with a complex model by using a relatively simple model would introduce errors.

To compensate for the prediction errors mentioned above, a tolerance parameter is introduced. As illustrated in figure 5-14, the tolerance parameter can be visualized as the radius of a prediction area. This prediction area is used in the observation step for detecting the target in this area in the next frame rather than localising the predictions to a single position.

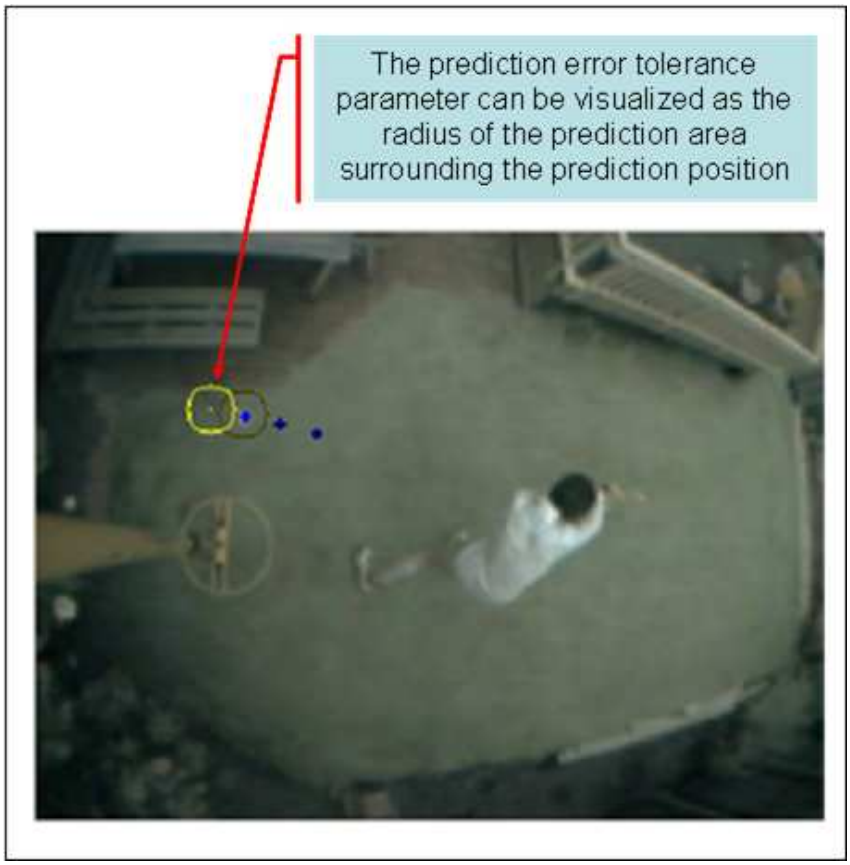


Figure 5-15: The tolerance parameter.

In the SOAPtet algorithm, the prediction error tolerance parameter is automatically adjusted to find the most suitable balance point between missing the targets and detecting false positives. Figure 5-16 illustrates a case where the prediction tolerance error is automatically adjusted due to the prediction error of an outward cricket ball target.

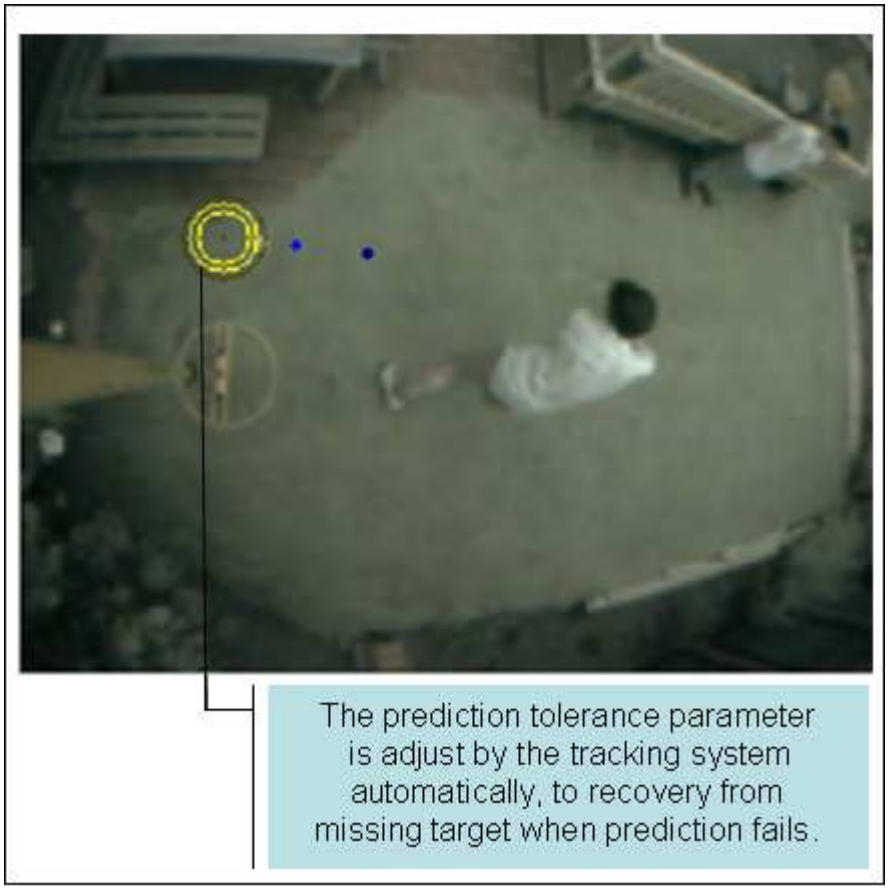


Figure 5-16: The tolerance value is adjusted by the system automatically.

5.3 Conclusion

The SOAPtet algorithm was presented in this chapter. In this algorithm, a DFSM is designed to constrain its behaviours in order to fit the target's motion properties at different motion states. Furthermore, a trilogy of observation, analysis and prediction is utilized to achieve the goals of target election and tracking. A novel score based target election approach is designed in this trilogy to separate the target of interest from candidates that are triggered by noise. Furthermore, a Kalman filter approach is utilised <change all US spelling to NZ in thesis> in the system to predict the trajectory properties of a target in the future frames. In addition, an automatically adjustable tolerance parameter is designed to compensate for prediction errors.

The SOAPtet algorithm is implemented in an experimental system that will be introduced in the next chapter. The implementation details and the experimental results will also be discussed in the next chapter.

Chapter 6

Experimental Evaluation

6.1 Introduction

The object detection and image segmentation algorithms and the target election and tracking algorithms have been discussed in chapter 4 and chapter 5. In this chapter, the accuracy, efficiency and robustness of these algorithms is evaluated through detection rate, tracking accuracy and efficiency.

A test system has been developed based on the PEDDA algorithm and the SOAPtet algorithm. This system is used to perform the experiments discussed in this chapter. The implementation issues are discussed in section 6.2

Three experiments have been carried out in section 6.3 using this test system, with two test video footages. These experiments evaluate detection rate, tracking accuracy and operating efficiency of the test system, and therefore, the PEDDA algorithm and the SOAPtet algorithm.

Finally, the key outcomes of these experiments are concluded in section 6.4

6.2 Test System Implementation

To evaluate the detection rate, tracking accuracy and running speed of the proposed algorithms, a test system that utilizes the algorithms that we discussed in chapter 4 and chapter 5 is implemented. This evaluation system is designed to test the algorithms by detecting and tracking cricket balls in real-time in unconstrained real world environments including variable luminance and camera motion.

The proposed test system will be introduced from both hardware and software perspectives. The hardware profile is discussed in section 6.2.1 and issues in the software implementation are discussed in section 6.2.2.

6.2.1 Hardware Perspective of the Test System

The hardware implementation of this system should fulfil the following three requirements:

- Support real-time image segmentation and objects tracking software.
- Provide appropriate real-time (30fps) video capture to the image segmentation and object tracking software.
- Clearly display results <remember we are not doing an HCI test which would evaluate effectiveness of presenting information to cricket players>.

This computer supported:

1. Strong computation capability and therefore can support real-time implementation of complex computer vision algorithms.
2. A sophisticated graphic rendering system, which can support presenting complex results.

This test platform is developed on a computer system that has the following profile:

- CPU: 1.54GHz AMD Athlon CPU
- RAM: 512MB
- Graphic Hardware: GeForce4 MX 440
- OS: Microsoft Windows XP Professional (Service Pack 2)
- USB 2 video capture.

In this research, a wireless camera (Figure 6-1 [a]) with 130 degrees lens is selected to provide appropriate real-time videos to the image segmentation and object tracking software. This camera is mounted on a mast at 3.5 meters height (Figure 6-1 [b]).

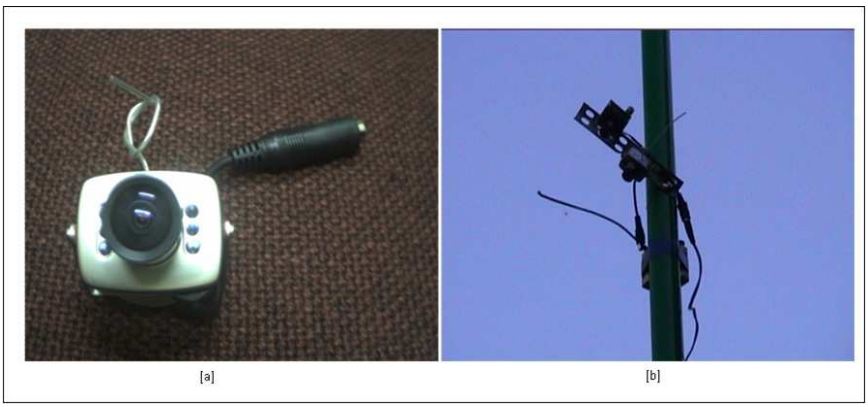
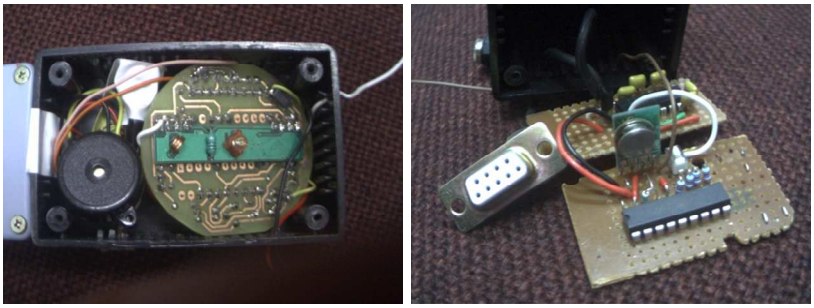


Figure 6-1 [a]: The wireless camera used in this project. [b]: the camera is mounted at 3.5 meters above ground level.

The parameters of this camera are listed below:

- Model: ZT-803
- Signal Transmission: Wireless radio frequency at 1.2GHz
- Transmission Range: about 30 meters
- Lens Angle: 130 degrees
- Power Supply: 6 x AAA battery pack (for 8? hours continuous transmission)
- Mount Position: 3.5 meters above ground level.
- 320 by 240 resolution 24 bit RGB colour image
- 30 frames pre second.

In this research, a wireless audio beeper is used to provide audio feedback information to the cricket players located remotely to the computer. Since it is difficult for a player to observe a computer's monitor in the test environment, this wireless audio beeper (Figure 6-2[a]) is chosen to prompt tracking results to the players. The transmitter part (Figure 6-2[b]) of this wireless audio beeper is connected to the computer via the serial port and the receiver part of this beeper is mounted on the top of the mast together with the wireless camera.



[a] [b]

Figure 6-2: Wireless audio beeper. [a]: The receiver and beeper; this part is mounted together with the camera at the top of the mast. [b]: The transmitter; this part is connected to the personal computer via a serial port.

6.2.2 Software Perspective of the Test System

6.2.2.1 The Software Modules

To support the objectives of this research with a tracking test system, the software implementation of this system should achieve the following five goals:

1. **Moving object detection and image segmentation**: Detect moving objects and segment these objects from the static background.
2. **Target election and tracking**: Elect the target among moving objects and tracking this target from frame to frame.
3. **Information exchange:** Provide a user interface to visualize information about the states of image segmentation and object tracking algorithms, such as the targets' trajectories and the system parameter values for test results. Also enable the parameters of this system to be conveniently adjusted through this interface.

4. **Video image acquisition:** Capture images from a live camera and video files to hard disk for subsequent analysis and comparison.
5. **Information Preservation:** Two categories of information need to be preserved. First, the parameters in this system should be saved to keep consistency between experiments. Second, the experimental results, such as the positions of beacons along a trajectory and an analysis time of each frame, should be saved for future analysis and reviewing.

To achieve the first two goals: moving object detection and tracking, the PEDDA algorithm and the SOAPtet algorithm that have been discussed in chapter 4 and chapter 5 are implemented in this test system.

A dialog based graphical user interface (GUI) is designed to achieve the third goal of information exchange. This dialog based GUI is shown in figure 6-3. The GUI has 4 main areas. The left top area contains 4 images. They are: the original image, the PEDDA result image, the blob information image and the trajectory beacon image. These images illustrate the current states of the PEDDA image segmentation algorithm and the SOAPtet object tracking algorithm. The left middle area contains several options that control the test system state. For example, the video source (video file or live camera) can be selected here. The left bottom area contains 7 control command buttons. These buttons are used to control this test system. On the right hand side is the parameters control panel. It lists 21 parameters that can affect object detection and target tracking algorithm running states, such as threshold values, detection area coordinates and so on. All these parameters are adjustable.

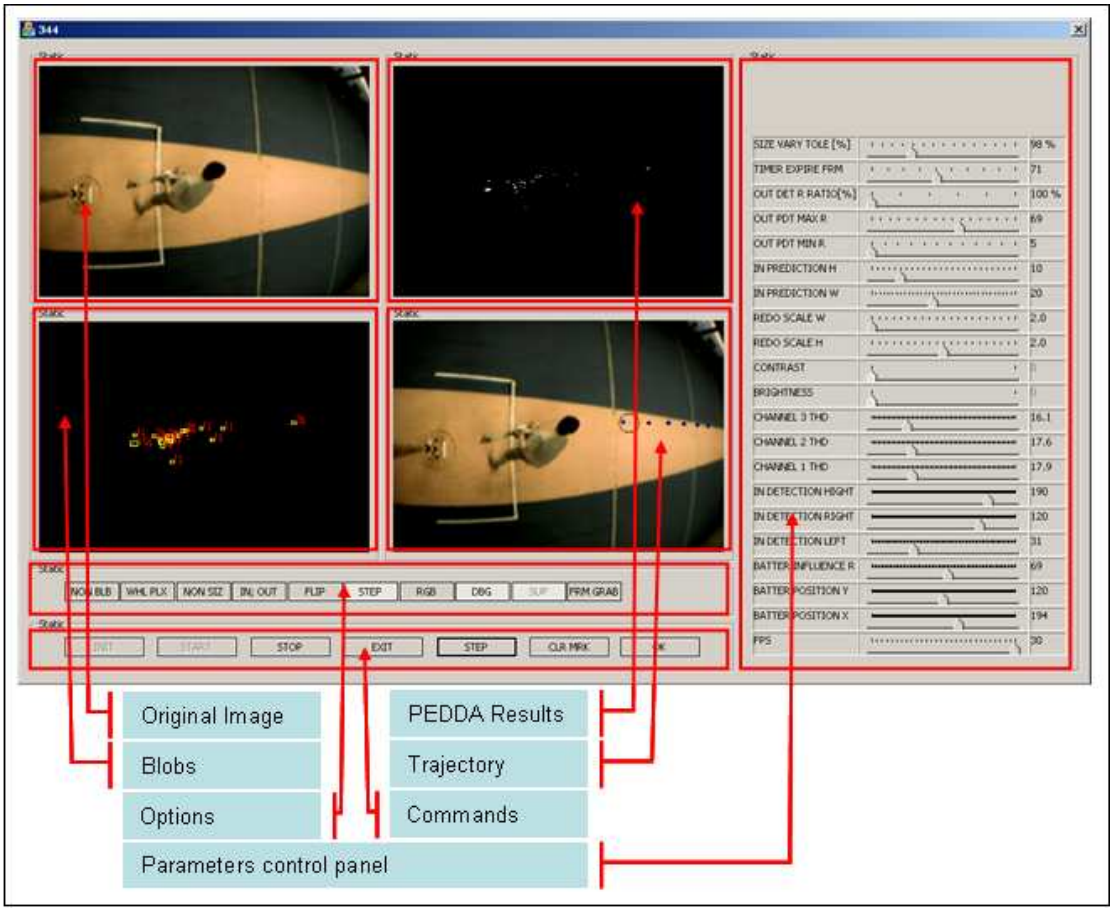


Figure 6-3: The graphical user interface of this test system.

To achieve the fourth goal: video image acquisition, this test system is implemented to be able to acquire images from both live camera and video files. The VidCapture [9] library is used in this system to capture live images from camera and the OpenCV [10] library is used to get images from video files.

Finally, registry and file system are used to achieve the fifth task: information preservation. There are 39 main parameters that need to be preserved. These parameters are stored in the operating system's registry. The CRegistry [11] library is used to carry on low level tasks of exchanging data between this test system and the operating system's registry. Apart from the parameters, the experimental data, such as

parsing frame rate and positions of each beacon along a trajectory are stored in the windows system as text files.

6.2.2.2 Multiple Threads Optimisation

To achieve a higher performance, this test system is implemented as a multithreaded application. Apart from the GUI thread, there are 3 other threads in this test system. They are the video capture thread, the moving object detection thread and the target tracking thread.

The video capture thread (Figure 6-4) grabs images from either a camera or a video file and puts the captured image into an image buffer that both this thread and the image segmentation thread can access to support asynchronous capture and processing.

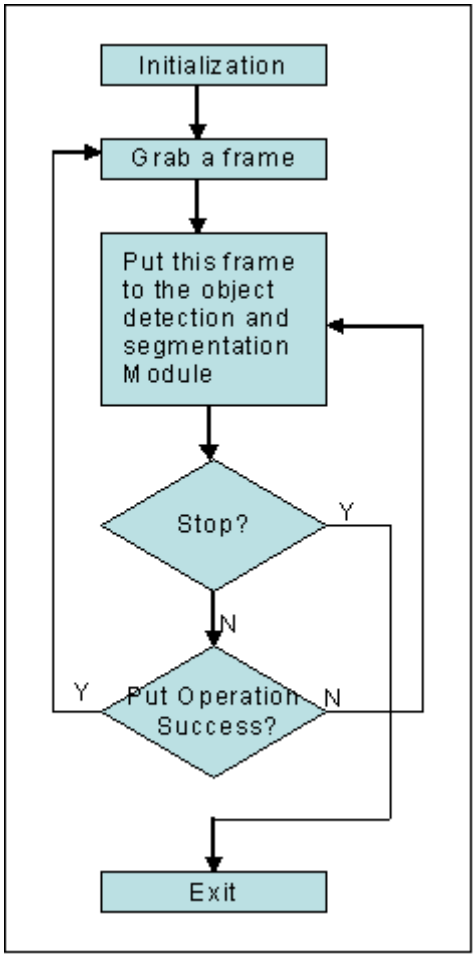


Figure 6-4: The video capture thread.

The moving object detection thread (Figure 6-5) provides a working environment for the PEDDA algorithm. It performs Gaussian smoothing and colour space conversion to the images and then utilizes the PEDDA algorithm to detect moving objects and segment the moving objects from a static background. Furthermore, this thread extracts the moving object's properties, such as centre point coordination and size, and filters these objects based on their properties. Finally, the target candidates are stored in a buffer that the target tracking thread can access.

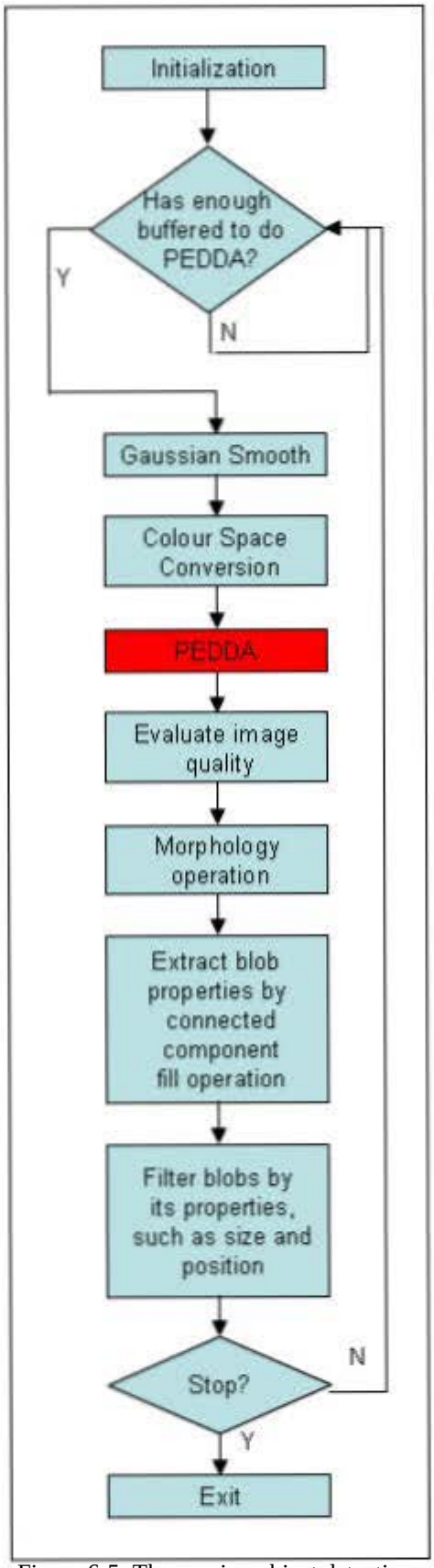


Figure 6-5: The moving object detection and image segmentation thread.

The SOAPtet algorithm is implemented in the object tracking thread (Figure 6-6). It grabs the moving blobs' information from the buffer that contains the results of the PEDDA algorithm and then feed these blobs' into the SOAPtet target election and tracking algorithm. The SOAPtet target election and tracing algorithm then elects the target out of all moving objects and tracks this target among frames.

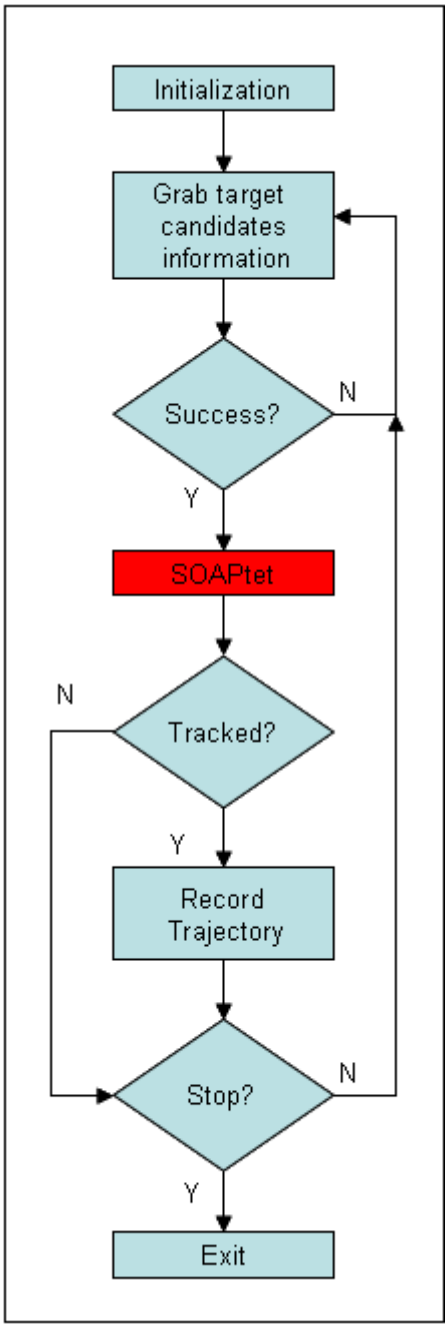


Figure 6-6: The target tracking thread.

6.3 Experiments

6.3.1 Introduction

In this section, the PEDDA moving objects detection and image segmentation algorithm and the SOAPtet target election and tracking algorithm will be evaluated with the test system developed in section 6.2.

Three experiments have been carried out to evaluate the developed moving object detection algorithm and target tracking algorithm. A target detection test and tracking accuracy experiment is designed to evaluate the quality of the proposed algorithms whereas the processing time experiment is designed for evaluation the efficiency of the developed algorithms.

Target detection test: As discussed in the chapter 5, in the context of the SOAPtet target election and tracking algorithm, a detection is defined as occurring when the state machine in this algorithm switching its state from S1 to S2 (the inward target cases) or from S3 to S4 (the outward target cases). Here, we define a successful detection as a state variation being triggered by a moving ball, either on the inward or outward direction. On the other hand, failure detections are defined as either detections that are triggered by noises (positive error) or a valid moving candidate failing to trigger the state variation (negative error).

In the target detection test, the successful detection rates are calculated with both inward and outward targets. Manual observation is involved in this test to classify the successful detections and failed detections. This experiment is discussed in detail in the section 6.3.2.

The tracking accuracy test: This experiment is about the mean deviation between the positions of beacons along a trajectory, calculated by the tracking algorithm and positions that are identified by a manual observation. A small value (near zero) of this mean deviation indicates the tracking algorithm is accurate. On the other hand, a large mean deviation value indicates that the tracking algorithm is inaccurate. In this test, the mean deviation value is calculated in the pixel units. To achieve a fair evaluation, the sample size of this experiment should be large enough, for example several hundreds samples. In the section 6.3.3, 620 beacons along trajectories of balls are researched and the statistical results are shown.

The processing time test: This experiment is designed to evaluate the efficiency of the proposed algorithms. The running speed of the system is indicated by the number of frames that the system can process in one second. The results of this experiment are closely related to the test platform where a faster computer system will increase the processing frame rate of a certain algorithm. The experimental results from two computer systems will be compared and discussed in the section 6.3.4.

Two video footages of ball sequences have been selected to perform the experiments mentioned above. Both of them were taken in real world environment by the low cost wireless camera that is introduced in the section 6.2.

The first video footage has 9016 frames and contains 62 inward and outward cricket balls. This video footage is taken in an outdoor environment with low luminance levels (Figure 6-7 [a]). Extraneous movement typical of a real-world environment is present as waving tree leaves and moving human bodies (Figure 6-7[b]) in this footage. Other real-world complexities exist in the form of target trajectory discontinuities such as bounces off obstacles in confined spaces (Figure 6-7[c]).

The second video footage has 7373 frames and contains 56 inward and outward cricket balls. This video footage is taken in a professional indoor environment with artificial lighting illumination (Figure 6-7[d]). Further real-world artefacts arise from ground reflections causing bright regions in this video footage (Figure 6-7[e]).The wireless radio transmission signal is likely to contain more noise in an indoor environment due to interference from closer electrical equipment (Figure 6-7[f]).

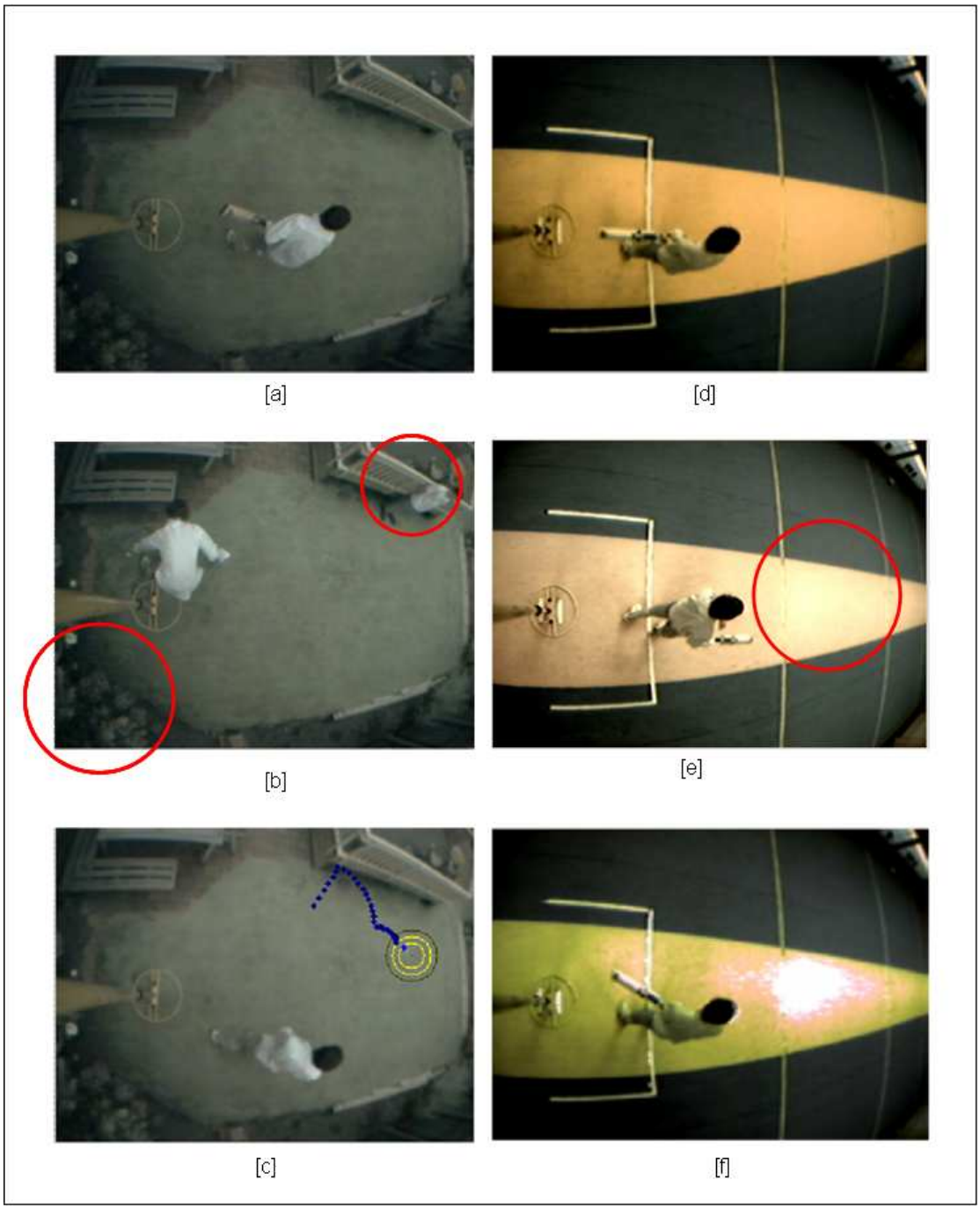


Figure 6-7: The video footages used in the experiments. [a] The first video footage is taken in a dark outdoor environment. [b] Moving objects (in the red circles) exist in the first video footage. [c] The targets have complex trajectories in the first video footage because of obstacles. [d] The second footage is taken in a professional indoor cricket place under well designed artificial illumination. [e] The bright area caused by ground reflection in the second footage is marked by red coloured circle in this image. [f] The image quality in this footage is lower that the first one because of the radio interference.

6.3.2 The Target Detection Test

The target detection test is done to evaluate the percentage of the cricket balls that can be detected by the PEDDA moving objects detection algorithm and be elected by the SOAPtet objects election and tracking algorithm.

Test Description:

Both the computer vision system and human subject are used in this target detection experiment. The object detection system is altered so that it is able to prompt a dialog (Figure 6-8) when it believes that there is an inward or outward cricket ball being detected. In this dialog, a number shows when (in terms of frame) this target is first detected. An experiment form (Figure 6-9) is required to be filled during this test by the human subject. This form is used to evaluate the test system's target detection results.

During the experiments, when a dialog is prompted to indicate a cricket ball is detected, the human subject may choose one of the two choices:

- If the human subject agrees that the algorithm's state is altered because of the appearance of a new moving object, such as a cricket ball, then she or he need to make a tick at the "In" or "Out" cells corresponding to the cricket ball's state and write down the frame number in the corresponding rows (Figure 6-9 [a]).
- If the human subject think that the state alteration of the algorithms is triggered by noise (Figure 6-10), then she or he need to make a tick at the "In

(+)" or "Out (+)" cells corresponding to the information writing on the dialog and write down the frame number in the corresponding rows (Figure 6-9[b]). The "In (+)" and "Out (+)" represent "Positive failures in detecting inward balls" and "Positive failures in detecting outward balls" respectively.

The computer vision based tracking system may fail to detect the cricket ball by negative errors too. Although the dialog will not be triggered in these cases, the missed cricket ball can be observed by the human participant. When the human participant notice that an inward and/or outward cricket ball is ignored by the tracking system, she or he should mark at the "In (-)" and/or "Out (-)" columns respectively just next to the last marked row (Figure 6-9 [c]). The "In (-)" and "Out (-)" represent "Negative failures in detecting inward balls" and "Negative failures in detecting outward balls" respectively.

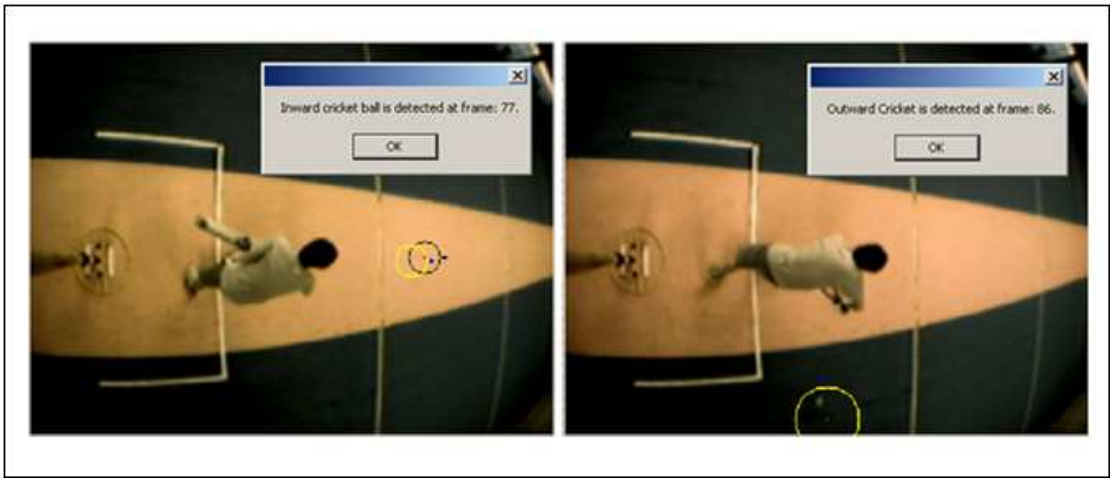


Figure 6-8: A dialog is prompted when the computer vision based system believes that a cricket ball, either inward or outward, is detected.

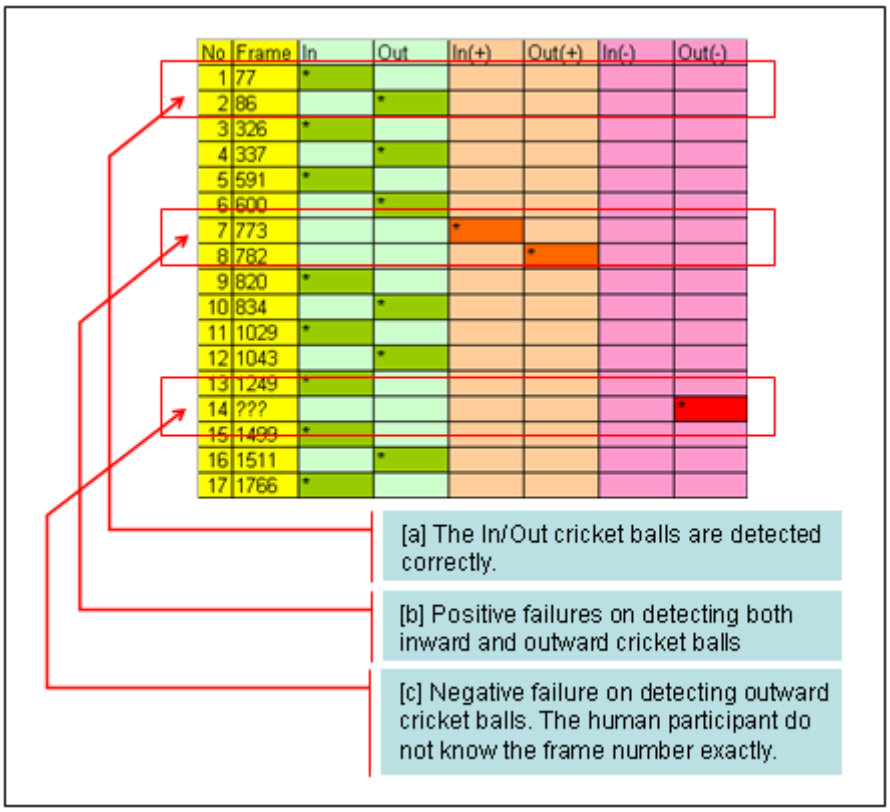


Figure 6-9: A form to be filled by the human participant. [a]: The first two rows are filled based on the information in Figure 6-8.

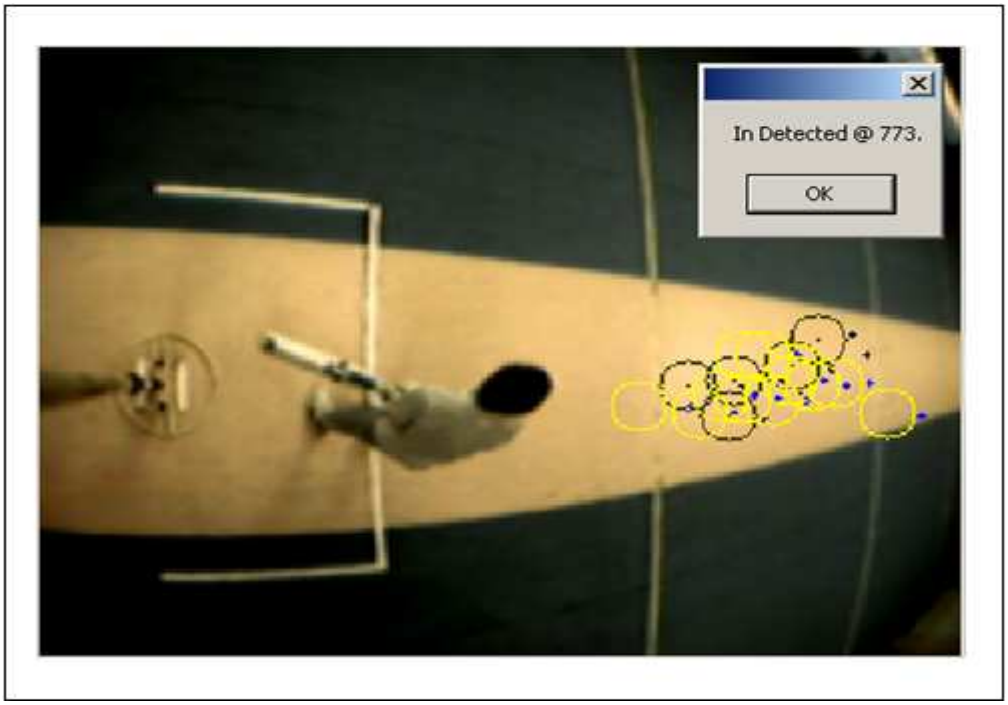


Figure 6-10: A case study of positive error. The dialog is triggered by noise, not an inward cricket ball.

Experimental Results:

The target detection test has been performed over both test video footages. The detailed experimental results are attached in the Appendix A.

Table 6-1 illustrates the main results obtained from the target detection test. In this table, the numbers of balls that have been observed by the human observer are listed in the "Balls Observed" column. The "Total detected" column lists the number of balls that the test system has detected Finally, the "False Negative" column lists the number of balls that has been observed by human subjects but has been ignored by the test system.

Video	Balls Observed	Total detected	Correct	False Positive	False Negative
1	62	62	58	4	4
2	56	55	51	4	5
Total	118	117	109	8	9

Table 6-1: the target detection experimental results

In video footage one, 93.5 percent of the cricket balls (58 out of 62) have been detected by the test system successfully. The rest of the balls (6.4 percent or four balls) are missed by the system (Figure 6-11 [a]). Apart from the total of 58 balls in the video footage one, the test system also reports four appearances of cricket balls. However, these four detection alerts cannot be verified by the human subject. These four balls are recorded as false positive cases. Thus rate of false positive in this video footage is 6.5% (Figure 6-11[b]).

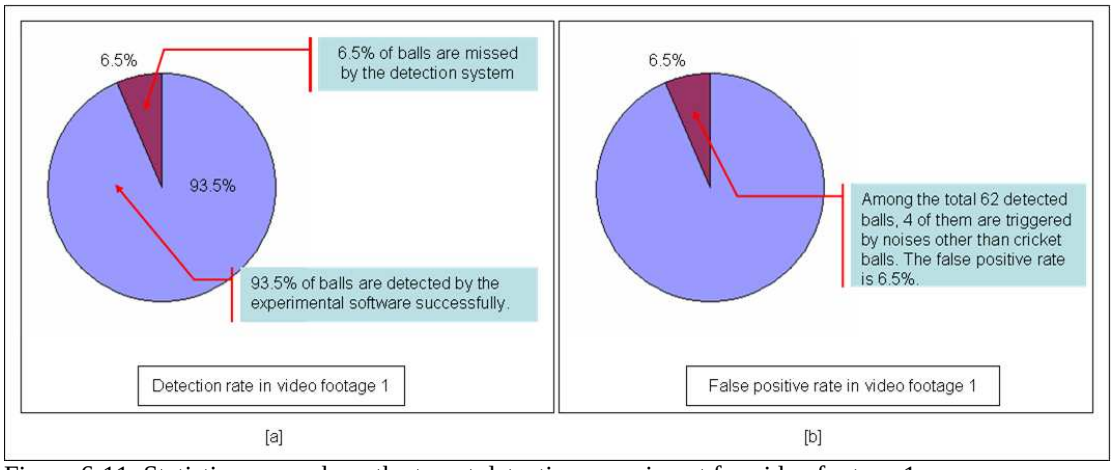


Figure 6-11: Statistics research on the target detection experiment for video footage 1.

In the video footage 2, 91.1 percent of balls were detected successfully (Figure 6-12[a]). Besides the misses balls, another 4 ghost balls (false positive) have been recorded by the human subject. The false positive rate is 7.3% (Figure 6-12[b]).

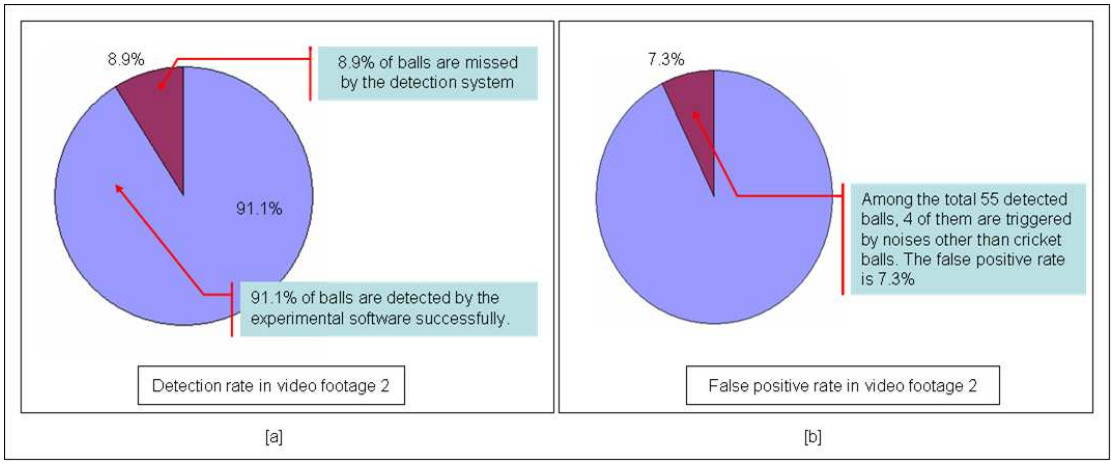


Figure 6-12: Statistics research on the target detection experiment for video footage 2.

Analysis:

The reasons that the system failed is commented on by the human subject based on her or his manually observation. There are two factors that can cause false positive

detection (ghost balls) to be observed by the human subject: noise and fast moving objects.

Although the algorithms that we proposed have in-built noise restrain measurements, these measurements cannot solve all of the noise problems. The extra pixel exclusion process in the PEDDA algorithm and the scoring scheme in the SOAPtet algorithm are designed to solve the noise problems and detect the tracked target even in the presents of high level noise. During the object detection experiments, these measurements have reduced a large amount of noise problems. However, in the worst cases, such as when the noise is caused by radio transmission error, these measurements are insufficient to recover the system. In the target detection test, 6.8% of the detections are false positive and 87.5% of the false positive errors are caused by noise problems.

The test system has reduced the moving objects problem successfully. Only one false positive error caused by the moving cricket pad has been observed. Among the total 8 ghost balls, only one case is triggered by the moving object problem (12.5%).

In addition, low contrast is the most significant factor that can cause false negative errors. There are two types of low contrast problems: low contrast caused by low illumination strength and low contrast caused by bright ground reflection area.

As mentioned in 6.3.1, one of the test video footage is taken in a dark environment. The contrast between the tracked target (a cricket ball) and ground maybe indistinguishable when the illumination strength is low. This type of low contrast

caused the system to miss 4 balls (50% of all false negative cases) in the target detection test.

The ground reflection problem usually happens in an indoor environment, especially under strong artificial illumination (Figure 6-7 e, f). A target may merge into the bright area and be hard to be observed by the test system. This type of low contrast caused another 4 cases of the false negative errors in the experiment.

6.3.3 Accuracy Test

The tracking accuracy test is carried out to evaluate the accuracy of the SOAPtet target election and tracking algorithm. The accuracy of the tracking results is defined as the distance between two positions: the position that the tracking system localized and the position that is localized by the human subject. The tracking error of a particular beacon point is defined as the distances between the human specified position and the position tracked by the test system. The human subject may introduce positioning error as well. However, the errors introduced by human factor are ignored in this experiment.

Test Description:

Both the computer vision system and a human are involved in the tracking accuracy test. An application (Figure 6-13 [a]) is developed for the human subject to localize the position of a cricket ball by using a 2D pointing devices, such as mouse. This

application records the pointing operations (Figure 6-13 [b]). This dedicated application is built to prevent the tracking information that is plotted on the test system from affecting the subject to point at a similar position as the tracking algorithm.

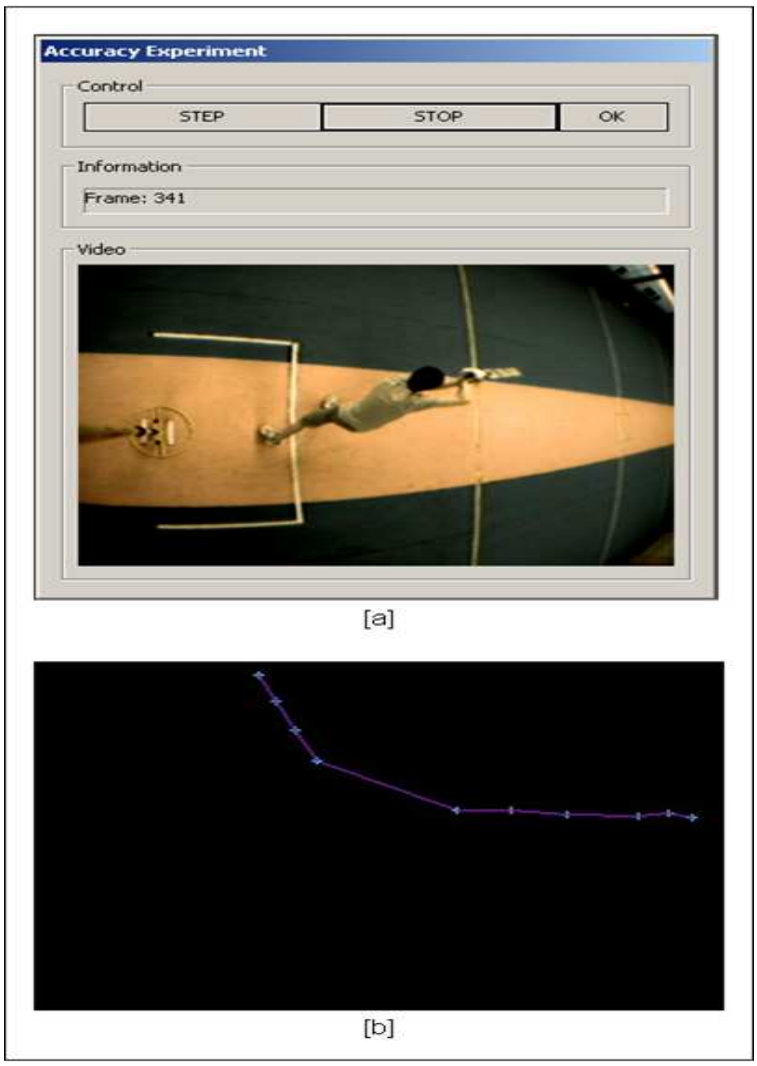
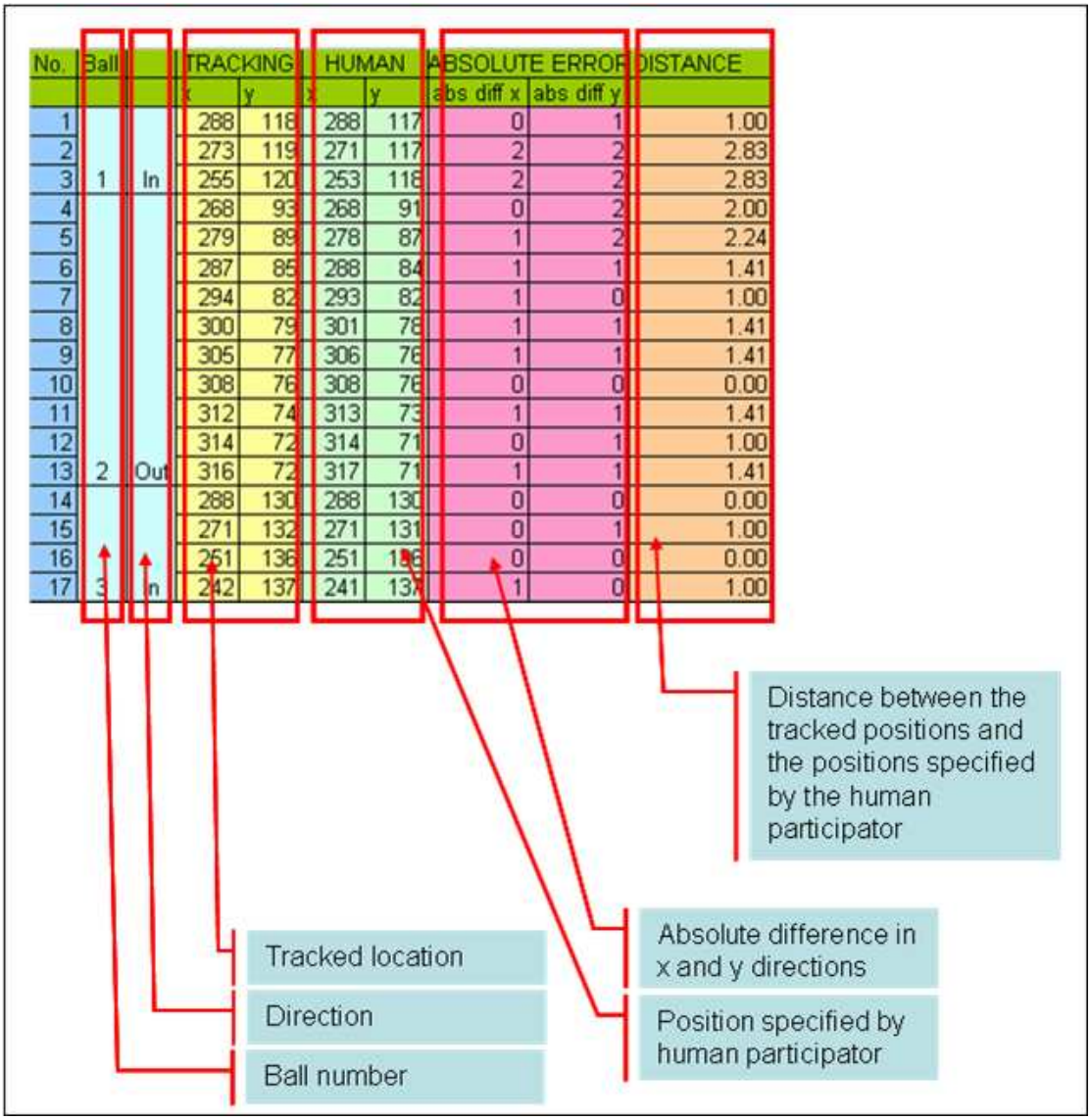


Figure 6-13: [a] An application to accept and record pointing operations. [b] The beacons along a trajectory that is recorded by this application.

The recorded data in this tracking accuracy test have been reformatted into a table like Figure 6-14. In this experiment, 620 beacons along 86 balls' trajectories have been researched. In Figure 6-14, the positions of these beacon points are listed in the

"TRACKING" column and the positions of the corresponding beacon points that are recorded from human subject are listed in the "HUMAN" column. The "ABSOLUTE ERROR" column records the absolute difference in both x and y directions between the two positions of beacon points. Finally, the "DISTANCE" column lists the distance in terms of pixels between the human subject specified positions and the tracking algorithm recorded positions.



No.	Ball		TRACKING		HUMAN		ABSOLUTE ERROR		DISTANCE
			x	y	x	y	abs diff x	abs diff y	
1			288	118	288	117	0	1	1.00
2			273	119	271	117	2	2	2.83
3	1	In	255	120	253	118	2	2	2.83
4			268	93	268	91	0	2	2.00
5			279	89	278	87	1	2	2.24
6			287	85	288	84	1	1	1.41
7			294	82	293	82	1	0	1.00
8			300	79	301	78	1	1	1.41
9			305	77	306	76	1	1	1.41
10			308	76	308	76	0	0	0.00
11			312	74	313	73	1	1	1.41
12			314	72	314	71	0	1	1.00
13	2	Out	316	72	317	71	1	1	1.41
14			288	130	288	130	0	0	0.00
15			271	132	271	131	0	1	1.00
16			251	136	251	136	0	0	0.00
17	3	In	242	137	241	137	1	0	1.00

Figure 6-14: the tracking accuracy experiment working sheet.

Experimental Results:

The tracking accuracy test has been done with 86 tracked trajectories. 620 beacon points have been recorded along these trajectories. The experimental data are shown in the Appendix A (A.1.2).

Table 6-2 illustrates the main statistics of the experimental data in this experiment. In this table, the mean, median, max, min and standard deviation values are calculated and listed. The sample space of these statistics studies are 620. The mean distance error in this experiment is 1.37 pixels. There are 97.42% of the tracking errors are less than or equal to 3 pixels (Figure 6-15).

No.	Function name	Subjects	Sample Size	Results
1	mean	absolute difference in x	620	0.68 pixel
2	mean	absolute difference in y	620	0.99 pixel
3	mean	distance	620	1.37 pixel
4	median	absolute difference in x	620	1 pixel
5	median	absolute difference in y	620	1 pixel
6	median	distance	620	1.41 pixels
7	max	absolute difference in x	620	15 pixels
8	max	absolute difference in y	620	4 pixels
9	max	distance	620	15.52 pixels
10	min	absolute difference in x	620	0 pixels
11	min	absolute difference in y	620	0 pixels
12	min	distance	620	0 pixels
13	standard deviation	absolute difference in x	620	1.08
14	standard deviation	absolute difference in y	620	0.68
15	standard deviation	distance	620	1.09

Table 6-2: the statistics study of the tracking accuracy experimental results.

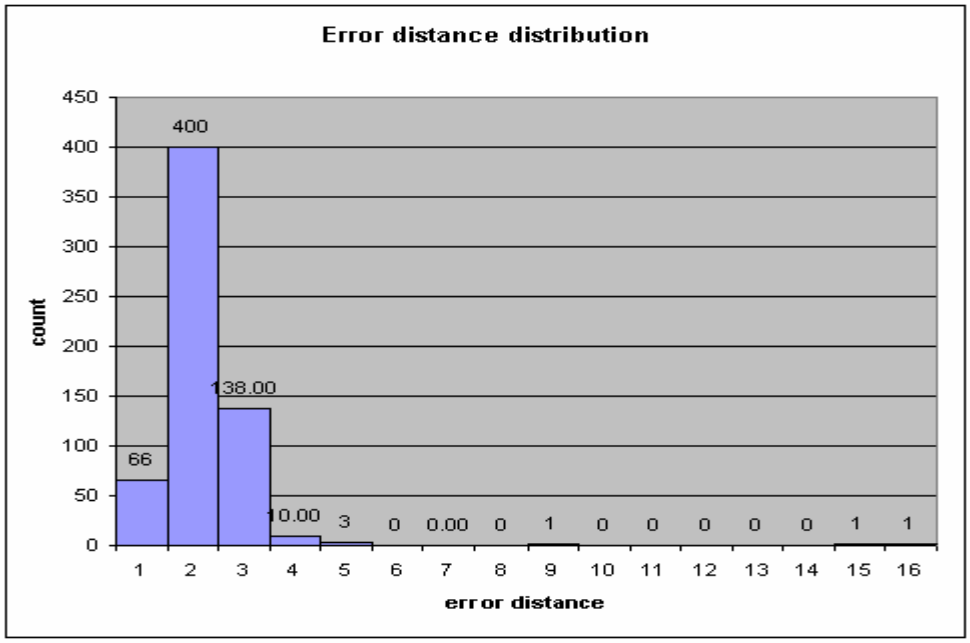


Figure 6-15: The distribution of tracking errors. 97.42% of errors are less than or equal to 3 pixels.

Analysis:

The average tracking errors between the test system and the human subject are less than 2 pixels. However, 3 beacon points that have large error distances (up to 15.52 pixels) have been recorded in this test. Apart from the effect of positioning errors involved by the human subject, the noise problem is the most significant source of tracking error. The figure 6-15 illustrates how noises affect the tracking accuracy in the SOAPtet tracking algorithm. In the first clean image, only the tracked target, a cricket ball, has been segmented from static background. The SOAPtet object tracking algorithm can localize the target accurately in this case. However, if the image is polluted by noises, for example in figure 6-15 [b], the test system cannot segment the tracked target clearly. Therefore, the SOAPtet tracking algorithm will use the mean position of all the foreground objects, which include the real target and noises, to

locate the tracked object. This technique helps the tracking system to survive in very noisy image environment but at the same time it reduces the tracking accuracy.

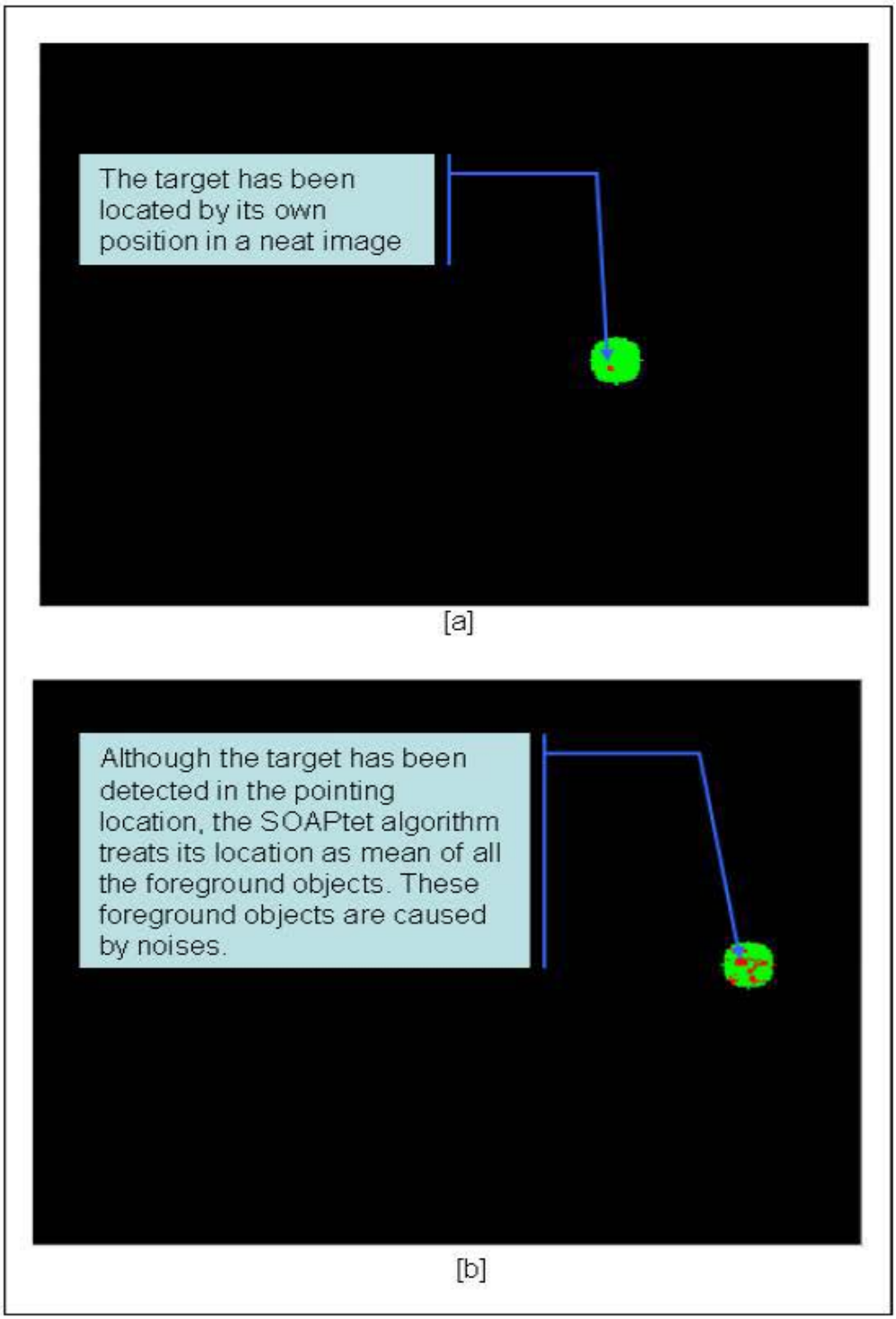


Figure 6-16: the noises effect tracking accuracy in this system. [a] the tracked target is segmented in a clean image. [b] The tracked target is segmented in an image that has been contaminated by noise.

6.3.4 Efficiency Test

The efficiency test is done to evaluate the speed of the proposed tracking system in terms of frame rate.

Test Description:

The efficiency test was carried out on two computer systems. Their profiles are listed in the table 6-3. To complete this experiment, the test system logs the number of frames it has processed every second. To achieve an accurate measurement on the object detection and tracking system, the graphic replay and algorithm visualization modules are isolated from the test system.

Computer No	CPU Model	CPU Speed	RAM
Computer 1	AMD	1.54 GHz	512M
Computer 2	INTEL	3.20 GHz	1G

Table 6-3: the two computers used in the efficiency experiment

Experimental Results:

The experimental data are listed in the Appendix A (A.1.3). A summary of these experimental results is listed in the table 6-4. The test results are shown in figure 6-17.

Video No	Computer No	Average Frame Rate
Test Video Footage 1	Computer 1	32.02 fps
Test Video Footage 2	Computer 1	32.52 fps
Test Video Footage 1	Computer 2	63.23 fps
Test Video Footage 2	Computer 2	63.65 fps

Table 6-4: a summary of efficiency test

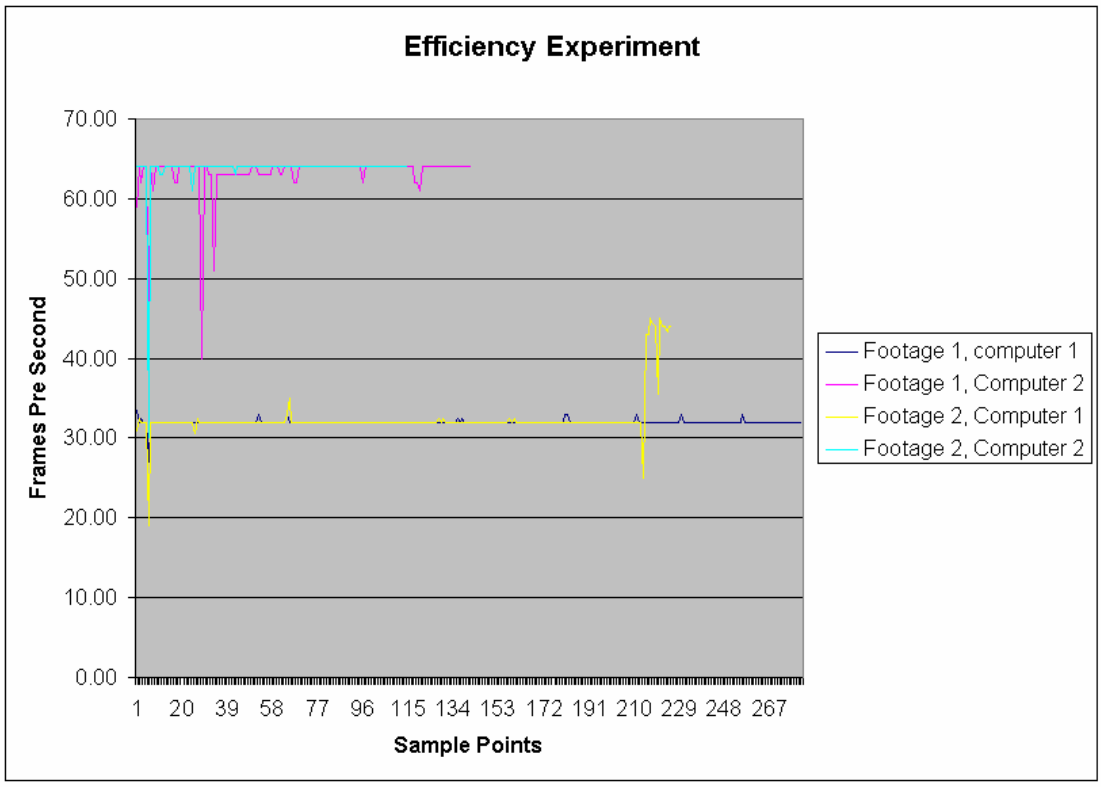


Figure 6-17: the Test results of efficiency experiment.

Analysis

The average frame rate of the first test video footage is faster that the second test video footage on both computer systems. This result is caused by the quality of the video footage. As the second test video footage has lower quality than the first one, more computing time have been wasted to readjust the threshold and prediction operations. Improve image quality can increase the processing speed in term of minimizing redo operations on both threshold adjusting and prediction.

6.4 Conclusion

Three experiments have been discussed in this chapter. These experiments evaluate detection rate tracking accuracy and processing efficiency of the test system developed based on the PEDDA algorithm and the SOAPtet algorithm. The experimental results are concluded as:

1. The object detection algorithm and target election algorithm achieve a high successful detection rate (more than 90 percent) and a low false detection rate (less than 10 percent in both false positive and false negative).
2. The tracking algorithm is precise. The average error in the tracking accuracy test is less than 2 pixels.
3. The test system is a real time system and can achieve more than 60 fps processing speed on a fast computer and more than 30 fps on a slower computer.
4. The noise is the most significant parameter that affects the detection rate, tracking accuracy and even efficiency of the system.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this research, a set of computer vision algorithms are designed to track small and fast objects in low quality image sequences. An experimental system is implemented based on the proposed PEDDA object detection algorithm and the proposed SOAPtet target election and tracking algorithm.

This novel PEDDA object detection algorithm is based on the general DDA algorithm. This algorithm suppresses low speed movement objects from being detected by analyzing the 4^{th} image frame in addition to the general DDA algorithm's three frames.

The novel SOAPtet target election and tracking algorithm utilizes a deterministic finite state machine to control the internal tracking strategies. A novel score based target election algorithm is embedded in the SOAPtet algorithm to elect the target of interest from candidates that are triggered by noise. In addition, a Kalman filter is inbuilt in the SOAPtet algorithm as well. The estimated position beacon in the proceeding frame along a trajectory is predicted through this Kalman filter.

Experiments are carried out with an experimental system implemention based on the two novel algorithms. The experimental results show that this experimental system detects 92.3% tracking targets successfully in noisy images. Furthermore, the accuracy experiment shows that 97.42% of tracking accuracy can be achieved. This accuracy experiment is based on statistical evaluation of over 620 beacon samples. Finally, the efficacy test results show that the system developed in this research is faster than real-time.

In a summary, two novel algorithms, the PEDDA object detection and the SOAPtet algorithm, are designed to fulfil the research motivations of:

- Detecting small diameter objects in a relative large search space.
- Detecting and tracking fast objects.
- Continuous tracking of objects with large scale trajectory discontinuities.

7.2 Future Work

This research can be improved in the following aspects:

- In this research, the PEDDA algorithm and the SOAPtet algorithm are implemented in a cricket ball tracking system. Further research can be done to utilize both algorithms in other computer vision based tracking systems, such

as video surveillance systems, human-computer interaction systems and entertainment systems.

- The deterministic finite state machine used in the SOAPtet algorithm is related to the target's trajectory feature and the control logic of the algorithm is set manually. The disadvantage of this is when the SOAPtet algorithm is used in another environment, some parameters have to be manually altered to fit the trajectory properties of the new tracking targets. Research can be done to build an automatic strategy establishing system to change the control logic of SOAPtet algorithm without manual intervention.
- Finally, the trajectory information is calculated based on the observation of a single camera. The limitation of this system is that only 2D trajectory can be captured. Future research can be carried out to utilize multiple cameras to calculate the 3D trajectories of the tracking targets.

References

[1] *Cai, Q.; Mitiche, A.; Aggarwal, J.K.;* **Tracking human motion in an indoor environment;** Image Processing, 1995. Proceedings., International Conference on , Volume: 1 , 23-26 Oct. 1995 Pages:215 - 218

[2] *Cucchiara, R.; Piccardi, M.; Prati, A.; Scarabottolo, N.;* **Real-time Detection of Moving Vehicles**; Image Analysis and Processing, 1999. Proceedings. International Conference on , 27-29 Sept. 1999 Pages:618 – 623

[3] *Toyama, K.; Krumm, J.; Brumitt, B.; Meyers, B.;* **Wallflower: Principles and practice of background maintenance;** Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on , Volume: 1 , 20-27 Sept. 1999 Pages:255 - 261

[4] *Cucchiara, R.; Grana, C.; Piccardi, M.; Prati, A.;* **Detecting moving objects, ghosts, and shadows in video streams;** Pattern Analysis and Machine Intelligence, IEEE Transactions on , Volume: 25 , Issue: 10 , Oct. 2003 Pages:1337 – 1342

[5] *Wang, L.; Hu, W.; Tan, T.;* **Recent developments in human motion analysis**; Pattern Recognition, 36(3):585–601, March 2003.

[6] *Horprasert, T.; Harwood , D.; Davis, L.;* **A Statistical Approach for Real Time Robust Background Subtraction and Shadow Detection**; IEEE Frame Rate Workshop 1999

[7] *Elgammal, A.; Harwood, D.; Davis, L. S.;* **Nonparametric background model for background subtraction**; In 6th European Conference of Computer Vision, 2000

[8] http://www.cmlab.csie.ntu.edu.tw/cml/dsp/training/coding/h263/s_coder.html, 17-Feb, 2005.

[9] http://www.codevis.com/, 17-Feb, 2005.

[10] http://www.intel.com/research/mrl/research/opencv/, 17-Feb, 2005.

[11] http://www.codeproject.com/system/caaregistryclass.asp, 17-Feb, 2005.

[12] *Hong, D; Woo, W;* **A Background Subtraction for a Vision-based User Interface;** Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia; Proceedings of the 2003 Joint Conference of the Fourth International Conference on , Volume: 1 , 15-18 Dec. 2003 Pages:263 - 267

[13] *Haritaoglu, I.; Harwood, D.; Davis, L.S.;* **W4: Who? When? Where? What? A Real Time System for Detecting and Tracking People;** Automatic Face and

Gesture Recognition, 1998; Proceedings; Third IEEE International Conference on 14-16 April 1998 Pages:222 - 227

[14] *Wren, C. R.; Azarbayejani, A.; Darrell, T.; Pentland, A.P.;* **Pfinder: Real-time Tracking of the Human Body.** Pattern Analysis and Machine Intelligence, IEEE Transactions on , Volume: 19 , Issue: 7 , July 1997 Pages:780 - 785

[15] *Horprasert, T.; Harwood, D.; Davis, L.;* **A Statistical Approach for Real Time Robust Background Subtraction and Shadow Detection;** IEEE Frame Rate Workshop 1999.

[16] http://www.physics.ohio-state.edu/~gan/teaching/spring04/Chapter3.pdf, 17-Feb, 2005.

[17] http://mathworld.wolfram.com/CentralLimitTheorem.html, 17-Feb, 2005.

[18] *McClave, T. J.;* **Statistics**. Prentice-Hall, Inc. Upper Saddle River, NJ 07458. 2000. pp 265-270

[19] http://mathworld.wolfram.com/Mean.html , 17-Feb, 2005.

[20] http://mathworld.wolfram.com/StandardDeviation.html, 17-Feb, 2005.

[21] *Stauffer, C.; Grimson, W. E. L.;* **Adaptive Background Mixture Models for Real-time Tracking.** Computer Vision and Pattern Recognition, 1999; IEEE Computer Society Conference on; Volume: 2 , 23-25 June 1999 Pages: 252

[22] *Wang, D.; Feng, T.; Shum, H. Y.; Ma, S. D.;* **A Novel Probability Model for Background Maintenance and Subtraction.** National Laboratory of Pattern Recognition, Chinese Academy of Science; Microsoft Research Asia, Beijing, China

[23] *Porikli, F.; Tuzel, O.;* **Human Body Tracking by Adaptive Background Models and Mean-Shift Analysis.** Mitsubishi Electric Research Laboratories, Inc., 201 Broadway, Cambridge, Massachusetts 02139, 2003

[24] *Gao, H.; Green R.;* **State-based Ball Detection and Tracking for Cricket Training**. In Proceedings of Image and Vision Computing New Zealand 2004, pages 423-428, Akaroa, New Zealand, 21-23 November 2004.

[25] *Lipton, A.; Fujiyoshi, H.; Patil, R.;* **Moving Target Classification and Tracking from Real-time Video**; in Proc. 4th IEEE Workshop on Application of Computer Vision (WACV '98), 1998, pp. 8–14.

[26] *Cucchiara, R.; Piccardi, M.; Mello, P.;* **Image Analysis and Rule-based Reasoning for a Traffic Monitoring System;** Intelligent Transportation Systems, IEEE Transactions on , Volume: 1 , Issue: 2 , June 2000 Pages:119 – 130

[27] *Baker, M.; Yanco, H. A.;* **A Vision-Based Tracking System for a Street-Crossing Robot**; UML Technical Report.

[28] http://en.wikipedia.org/wiki/Finite_state_machine, 17-Feb, 2005.

[29] http://faculty.kutztown.edu/rieksts/126/fsm/basics.html, 17-Feb, 2005.

[30] *Gibson, D.;* **Making Simple Work of Complex Functions**. SPLat Controls Pty Ltd. 1998-2000

[31] *Welch, G.; Bishop, G.;* **An Introduction To the Kalman Filter**; TR 95-041, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599-3175. April, 2004

[32] *Kalman, R. E.;* **A New Approach to Linear Filtering and Prediction Problems**, Transaction of the ASME – Journal of Basic Engineering, pp. 35-45 (March 1960)

[33] *Kay, S. M.;* **Fundaments of Statistical Signal Processing,** Volume 1: Estimation Theory. Prentice Hall, 1993

[34] *Wang, J.; Xu, C. S.; Chng, E. S.; Wah, K.; Tian, Q.;* **Automatic Replay Generation for Soccer Video Broadcasting**; Proceedings of the 12th annual ACM international conference on Multimedia, October 10-16, 2004, New York, NY, USA

[35] *Ren, J.; Orwell, J.; Jones, G.A.; Xu, M.;* **A General Framework for 3D Soccer Ball Estimations and Tracking**; IEEE International Conference on Image Processing, October, Suntec City, Singapore.

[36] *D'Orazio, T.; Ancona, N.; Cicirelli, G.; Nitti, M.;* **A Ball Detection Algorithm for Real Soccer Image Sequences.** Pattern Recognition 2002. Proceedings 16th International Conference on , Volume: 1 , 11-15 Aug. 2002 Pages:210 - 213

[37] *Janez P,; Goran V.; Stanislav K.; Branko D.;* **A Low-Cost Real-Time Tracker of Live Sport Events**., University of Ljubljana, SI-1000 Ljubljana, Slovenija

[38] *Pingali, G. S.; Jean, Y.; Carlbom, I.;* **Real Time Tracking for Enhanced Tennis Broadcasts;** Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974

[39] *Wedge, D.; Huynh, D.; Koves, P.;* **Tracking Footballs Through Clutter in Broadcast Digital Videos;** In Proceedings of Image and Vision Computing New Zealand 2004, pages 155-160, Akaroa, New Zealand, 21-23 November 2004

[40] *Koschan, A.; Kang, S.; Paik, J.; Abidi, B.; Abidi, M.;* **Colour Active Shape Models for Tracking Non-rigid Objects.** Imaging, Robotics, and Intelligent Systems Laboratory, University of Tennessee, 330 Ferris Hall, Pattern Recognition Letters 24 (2003) 1751–1765

[41] http://mathworld.wolfram.com/ConfidenceInterval.html, 17-Feb, 2005.

[42] http://mathworld.wolfram.com/NormalDistribution.html, 17-Feb, 2005.

[43] *Pavlidis, I.; Morellas, V.; Tsiamyrtzis, P.; Harp, S.;* **Urban Surveillance Systems: from the Laboratory to the Commercial World.** Proceedings of the IEEE , Volume: 89 , Issue: 10 , Oct. 2001 Pages:1478 – 1497

[44] *Zhao, T.; Nevatia, R.; Lv, F.;* **Segmentation and Tracking of Multiple Humans in Complex Situations;** Computer Vision and Pattern Recognition, 2001. CVPR 2001; Proceedings of the 2001 IEEE Computer Society Conference on , Volume: 2 , 8-14 Dec. 2001 Pages:II-194 - II-201

[45] *Friedman, N.; Russell, S.;* **Image Segmentation in Video sequences: A Probabilistic Approach,** in Uncertainty in Articial Intelligence, 1997.

[46] *Grimson, W. E. L.; Stauer, C.;* Romano, R.; **Using Adaptive Tracking to Classify and Monitor Activities in a Site**, in CVPR, 1998.

[47] *Pers. J.; Kovacic. S.;* **Computer Vision System for Tracking Players in Sports Games**; First Int'l Workshop on Image and Signal Processing and Analysis, June 14-15, 2000, Pula, Croatia

[48] *Vaghani. K..; Green R.;* **Real-time Motion Capture in Public Spaces Using Stereo Vision**; In Proceedings of Image and Vision Computing New Zealand 2004, pages 143-148, Akaroa, New Zealand, 21-23 November 2004.

[49] *Lipton, A. .J.; Fujiyoshi, H.; Patil, R.S.;* **Moving Target Classification and Tracking from Real-time Video.** Applications of Computer Vision, 1998; WACV '98; Proceedings Fourth IEEE Workshop on , 19-21 Oct. 1998 Pages:8 – 14

[50] *Shin. J.;* **Initialization of Visual Object Tracker using Frame Absolute Difference.** Technical report; Computer Graphics Lab. Stanford University, CA 94305

[51] *Whitehead, A.;* **Fast Feature-based Video Segmentation and Annotation.** Signal Processing and Its Applications, 2003; Proceedings Seventh International Symposium on , Volume: 1 , 1-4 July 2003 Pages:637 - 640

[52] *Snyder, R..D.; Forbes, C. S.;* **Understanding the Kalman Filter: an Object Oriented Programming Perspective;** Monash Econometrics and Business Statistics Working Papers 14/99, Monash University, Department of Econometrics and Business Statistics.

[53] *Forsyth, D.; Ponce, J.;* **Computer Vision : a Modern Approach**.; Upper Saddle River, N.J. ; London : Prentice Hall, c2003, pp 373-397.

[54] *Mayeck, P. S.;* **Stochastic Models, Estimation, and Control**. Volume 1 Academic Press; New York. 1979. pp 1- 16.

[55] *Funk, N.;* **A study of the Kalman Filter applied to Visual Tracking**. Project report CMPUT 652, University of Alberta. Dec 7, 2003.

[56] *Wren., C., R.; Pentland, A., P.;* **Dynamic Models of Human Motion**; Proc. Third IEEE Intl. Conf. Automatic Face and Gesture Recognition, April 1998.

[57] *Beymer, D.; Konolige, K.;*. **Real-time Tracking of Multiple People Using Continuous Detection**.; Artificial Intelligence Center, SRI International Report, 1998.

[58] *Oliver, N, Rosario, B.; Pentland, A.;* **A Bayesian Computer Vision System for Modelling Human Interactions.** MIT Media Laboratory Report, 1998

[59] *Rosales, R.; Sclaroff, S.;* **3D Trajectory Recovery for Tracking Multiple Objects and Trajectory Guided Recognition of Actions**; Proc. of IEEE Conf. On Computer Vision and Pattern Recognition, June 1999.

[60] *Azarbayejani. A.; Pentland. A;* **Real-time Self-Calibrating Stereo Person Tracking Using 3D Shape Estimation from Blob Features**; Proc. of ICPR, IEEE, 1996

[61] *Bregler, C.;* **Learning and Recognizing Human Dynamics in Video Sequences**.; IEEE Press, pp 568-574, 1997.

[62] *Bodor, R., Jackson, B.; Papanikolopoulos, N.;* Vision-**Based Human Tracking and Activity Recognition**, Proc. of the 11th Mediterranean Conf. on Control and Automation, June 18-20, 2003.

[63] *Needham, J. C.; Boyle, D. R.;* **Tracking Multiple Sports Players Through Occlusion Congestion and Scale**. School of Computing, The University of Leeds., Leeds, LS2 9JT, UK

[64] *Kahl, F.; Hartley, R.; Hilsenstein, V.;* **Novelty Detection in Image Sequences with Dynamic Background**. Proc. 2nd Workshop on Statistical Methods in Video Processing (ECCV'04), Prague, Czech Republic, 2004

[65] *Siebel, T. N.; Maybank, S.;* **Fusion of Multiple Tracking Algorithms for Robust People Tracking;** In Proceedings of the 7th European Conference on Computer Vision (ECCV 2002), København, Denmark, May 2002, volume IV, pages 373--387, Springer Verlag, 2002, ISBN 3-540-43748-7.

[66] *Malley P. D.; Nechyba., N. C.; Arroyo, A. A.;* **Human Activity Tracking for Wide-Area Surveillance**; 2002 Florida Conference on Recent Advances in Robotics, Miami, May 2002.

[67] *Iannizzotto, G.; Villari, M.; Vita, L.;* **Hand Tracking for Human-Computer Interaction with *Graylevel VisualGlove*: Turning Back to the Simple Way;** ACM International Conference Proceeding Series**,** Proceedings of the 2001 workshop on Perceptive user interfaces, Orlando, Florida POSTER SESSION: Posters & demos. pp: 1 – 7, 2001

[68] *Freeman, W.T.; Tanaka, K.; Ohta, J.; Kyuma, K.;* **Computer Vision for Computer Games.** Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on , 14-16 Oct. 1996 Pages:100 - 105

Appendix A

A.1 Experiment Results

A.1.1 Target Detection Experiment Results

Table A-1 and table A-2 illustrate the whole experimental results that are acquired from the target detection experiments mentioned in the section 6.3.2

Video: Test Video 1								
Total Frames: 9016								
No	**Frame**	**In**	**Out**	**In(+)**	**Out(+)**	**In(-)**	**Out(-)**	**Comments**
1	56	*						
2	67		*					
3	215	*						
4	233		*					
5	331	*						
6	353		*					
7	601	*						
8	617		*					
9	922	*						
10	935		*					
11	1172	*						
12	1182		*					
13	1199			*				Noises casused by radio transmission error
14	1227				*			
15	1470	*						
16	1489		*					
17	1911	*						
18	1925		*					
19	2040	*						
20	2055		*					
21	2205			*				Noises casused by radio transmission error
22	2282	*						
23	2302		*					
24	2403	*						
25	2421		*					
26	???					*		Low contrast object in dark environment
27	???						*	
28	2993	*						
29	3005		*					
30	3158	*						
31	3184		*					
32	3280	*						
33	3297		*					
34	3827	*						
35	3848		*					

36	3860			*				Noises casused by radio transmission error
37	3962	*						
38	3981		*					
39	4470	*						
40	4485		*					
41	4858	*						
42	4870		*					
43	4966	*						
44	4974		*					
45	5580	*						
46	5594		*					
47	6171	*						
48	6197		*					
49	6564	*						
50	6576		*					
51	6849	*						
52	6860		*					
53	7102	*						
54	7114		*					
55	7317	*						
56	7330		*					
57	7781	*						
58	7795		*					
59	8074	*						
60	8089		*					
61	8402	*						
62	8423		*					
63	???					*		Low contrast object in dark environment
64	???						*	
65	8966	*						
66	8970		*					

Table A-1: Experimental result in the target detection experiment (Video footage 1).

Video: Test Video 2								
Total Frames: 7373								
No	**Frame**	**In**	**Out**	**In(+)**	**Out(+)**	**In(-)**	**Out(-)**	**Comments**
1	77	*						
2	86		*					
3	326	*						
4	337		*					
5	591	*						
6	600		*					
7	773			*				Noises casused by radio transmission error
8	782				*			
9	820	*						
10	834		*					
11	1029	*						

12	1043		*					
13	1249	*						
14	???						*	High light area cased by ground reflection
15	1499	*						
16	1511		*					
17	1766	*						
18	1781		*					
19	1946			*				Noises casused by radio transmission error
20	1969	*						
21	1982		*					
22	2266	*						
23	2282		*					
24	2459	*						
25	???						*	High light area cased by ground reflection
26	2972	*						
27	2987		*					
28	3252	*						
29	3271		*					
30	3484	*						
31	3502		*					
32	3926	*						
33	3944		*					
34	4111	*						
35	4126		*					
36	4325	*						
37	4333		*					
38	4462	*						
39	4481		*					
40	4716	*						
41	4734		*					
42	4953	*						
43	4970		*					
44	5323	*						
45	5340		*					
46	5523	*						
47	5540		*					
48	5775	*						
49	???						*	High light area cased by ground reflection
50	6074	*						
51	6085		*					
52	???					*		High light area cased by ground reflection
53	???						*	
54	6765	*						
55	6779		*					
56	7005	*						
57	7019		*					
58	7261	*						
59	7274		*					
60	7293			*				Moving Cricket pad.

Table A-2: Experimental result in the target detection experiment (Video footage 2)

A.1.2 Tracking Accuracy Experiment

No.	Ball		TRACKING		HUMAN		ABSOLUTE ERROR		DISTANCE
			x	y	x	y	abs diff x	abs diff y	
1	1	In	288	118	288	117	0	1	1.00
2			273	119	271	117	2	2	2.83
3			255	120	253	118	2	2	2.83
4	2	Out	268	93	268	91	0	2	2.00
5			279	89	278	87	1	2	2.24
6			287	85	288	84	1	1	1.41
7			294	82	293	82	1	0	1.00
8			300	79	301	78	1	1	1.41
9			305	77	306	76	1	1	1.41
10			308	76	308	76	0	0	0.00
11			312	74	313	73	1	1	1.41
12			314	72	314	71	0	1	1.00
13			316	72	317	71	1	1	1.41
14	3	In	288	130	288	130	0	0	0.00
15			271	132	271	131	0	1	1.00
16			251	136	251	136	0	0	0.00
17			242	137	241	137	1	0	1.00
18	4	Out	248	93	248	93	0	0	0.00
19			258	82	257	82	1	0	1.00
20			262	76	261	76	1	0	1.00
21			265	73	265	73	0	0	0.00
22			267	69	267	67	0	2	2.00
23			269	64	269	65	0	1	1.00
24			270	62	270	61	0	1	1.00
25			270	58	270	57	0	1	1.00
26			269	55	269	56	0	1	1.00
27			269	53	267	52	2	1	2.24
28			267	51	267	50	0	1	1.00
29			266	50	265	49	1	1	1.41
30			264	48	263	49	1	1	1.41
31			262	47	261	47	1	0	1.00
32			259	46	258	45	1	1	1.41
33			256	45	256	44	0	1	1.00
34			253	45	253	45	0	0	0.00
35			250	45	249	45	1	0	1.00
36			247	45	246	44	1	1	1.41
37			244	45	243	45	1	0	1.00
38			239	45	240	45	1	0	1.00
39			239	44	238	43	1	1	1.41
40			241	40	241	40	0	0	0.00
41	5	In	316	117	316	116	0	1	1.00
42			306	118	306	116	0	2	2.00
43			294	120	293	118	1	2	2.24
44			280	120	280	119	0	1	1.00
45			263	122	264	121	1	1	1.41
46	6	Out	130	83	130	82	0	1	1.00

47			103	80	102	79	1	1	1.41
48			90	78	90	77	0	1	1.00
49			69	77	66	76	3	1	3.16
50			56	77	57	75	1	2	2.24
51			48	77	46	75	2	2	2.83
52			40	77	40	77	0	0	0.00
53			33	78	32	77	1	1	1.41
54			25	77	24	77	1	0	1.00
55			18	75	18	75	0	0	0.00
56			11	73	11	72	0	1	1.00
57			4	71	3	70	1	1	1.41
58	7	In	312	113	312	112	0	1	1.00
59			298	114	298	114	0	0	0.00
60			284	115	284	114	0	1	1.00
61			266	116	267	115	1	1	1.41
62	8	Out	253	111	253	110	0	1	1.00
63			272	109	272	107	0	2	2.00
64			280	108	280	108	0	0	0.00
65			293	106	293	105	0	1	1.00
66			298	106	299	105	1	1	1.41
67			302	106	302	105	0	1	1.00
68			306	105	306	105	0	0	0.00
69			317	104	317	103	0	1	1.00
70	9	In	284	105	285	103	1	2	2.24
71			267	105	266	105	1	0	1.00
72			247	105	246	104	1	1	1.41
73	10	Out	114	83	114	82	0	1	1.00
74			99	79	100	78	1	1	1.41
75			85	76	84	75	1	1	1.41
76			70	73	70	72	0	1	1.00
77			55	70	54	70	1	0	1.00
78			41	68	40	68	1	0	1.00
79			28	66	27	65	1	1	1.41
80			16	65	16	64	0	1	1.00
81			6	64	5	64	1	0	1.00
82	11	In	292	118	292	116	0	2	2.00
83			276	119	276	117	0	2	2.00
84			258	121	258	119	0	2	2.00
85			237	123	237	121	0	2	2.00
86	12	Out	200	65	200	64	0	1	1.00
87			199	61	198	61	1	0	1.00
88			198	57	198	57	0	0	0.00
89			196	54	195	54	1	0	1.00
90			194	51	194	50	0	1	1.00
91			192	49	192	48	0	1	1.00
92			189	46	189	45	0	1	1.00
93			188	44	189	43	1	1	1.41
94			186	40	186	40	0	0	0.00
95			185	36	184	34	1	2	2.24
96			183	32	182	31	1	1	1.41
97			181	28	181	27	0	1	1.00
98			179	24	178	23	1	1	1.41
99			176	21	176	19	0	2	2.00

100			172	18	173	17	1	1	1.41
101			169	15	169	15	0	0	0.00
102			167	12	166	11	1	1	1.41
103			163	10	162	9	1	1	1.41
104			159	8	158	7	1	1	1.41
105			155	7	155	6	0	1	1.00
106			151	6	150	4	1	2	2.24
107			148	4	148	2	0	2	2.00
108	13	In	316	113	315	112	1	1	1.41
109			303	114	303	113	0	1	1.00
110			288	115	288	114	0	1	1.00
111			270	115	270	114	0	1	1.00
112			248	117	249	116	1	1	1.41
113	14	Out	260	118	260	117	0	1	1.00
114			274	118	276	117	2	1	2.24
115			280	118	280	117	0	1	1.00
116			285	118	285	116	0	2	2.00
117			289	118	289	117	0	1	1.00
118			292	118	292	117	0	1	1.00
119			294	118	294	116	0	2	2.00
120			297	118	298	116	1	2	2.24
121	15	In	284	129	284	128	0	1	1.00
122			282	129	282	128	0	1	1.00
123			279	130	278	129	1	1	1.41
124			276	131	276	130	0	1	1.00
125			271	132	270	132	1	0	1.00
126			267	133	267	132	0	1	1.00
127			261	134	261	133	0	1	1.00
128			254	136	253	135	1	1	1.41
129	16	Out	265	130	265	130	0	0	0.00
130			270	129	269	128	1	1	1.41
131			274	128	273	128	1	0	1.00
132			277	126	276	124	1	2	2.24
133			280	125	280	123	0	2	2.00
134			282	124	282	123	0	1	1.00
135			282	124	283	122	1	2	2.24
136	17	In	292	110	292	110	0	0	0.00
137			276	110	277	109	1	1	1.41
138			263	110	263	109	0	1	1.00
139			259	110	258	109	1	1	1.41
140	18	Out	260	118	259	116	1	2	2.24
141			271	120	270	120	1	0	1.00
142			281	122	281	121	0	1	1.00
143			290	123	290	123	0	0	0.00
144			298	124	298	125	0	1	1.00
145			304	126	304	125	0	1	1.00
146			310	127	310	128	0	1	1.00
147			316	128	316	128	0	0	0.00
148	19	In	316	112	315	111	1	1	1.41
149			305	112	305	112	0	0	0.00
150			290	114	291	113	1	1	1.41
151			273	116	273	115	0	1	1.00
152			251	118	251	117	0	1	1.00

153	20	Out	260	123	260	121	0	2	2.00
154			280	122	278	121	2	1	2.24
155			289	122	288	120	1	2	2.24
156			302	122	302	120	0	2	2.00
157			308	122	308	121	0	1	1.00
158	21	In	296	124	295	123	1	1	1.41
159			282	126	281	125	1	1	1.41
160			276	128	276	129	0	1	1.00
161			274	129	274	128	0	1	1.00
162			271	129	271	129	0	0	0.00
163			267	130	266	130	1	0	1.00
164			263	131	264	131	1	0	1.00
165	22	Out	197	60	196	58	1	2	2.24
166			201	54	201	53	0	1	1.00
167			205	48	204	48	1	0	1.00
168			208	43	207	42	1	1	1.41
169			212	38	212	37	0	1	1.00
170			214	34	213	34	1	0	1.00
171			215	30	215	29	0	1	1.00
172			220	32	219	32	1	0	1.00
173			224	35	224	34	0	1	1.00
174			228	39	228	38	0	1	1.00
175			231	43	230	41	1	2	2.24
176			235	47	235	45	0	2	2.00
177			237	51	236	49	1	2	2.24
178			239	55	238	55	1	0	1.00
179			241	60	242	59	1	1	1.41
180			242	64	242	63	0	1	1.00
181			243	68	243	66	0	2	2.00
182			243	72	242	71	1	1	1.41
183			245	76	245	75	0	1	1.00
184			248	76	248	75	0	1	1.00
185			250	77	249	76	1	1	1.41
186			253	78	253	77	0	1	1.00
187			255	80	255	80	0	0	0.00
188			256	80	256	79	0	1	1.00
189			258	82	258	81	0	1	1.00
190			259	84	259	82	0	2	2.00
191			260	84	260	82	0	2	2.00
192			260	88	259	86	1	2	2.24
193	23	In	283	127	282	127	1	0	1.00
194			281	128	280	128	1	0	1.00
195			279	129	278	128	1	1	1.41
196			275	131	274	130	1	1	1.41
197			271	132	270	132	1	0	1.00
198			266	134	266	132	0	2	2.00
199			259	136	258	134	1	2	2.24
200			251	138	250	138	1	0	1.00
201	24	Out	256	174	256	173	0	1	1.00
202			287	181	286	180	1	1	1.41
203			300	184	299	182	1	2	2.24
204	25	In	302	124	302	125	0	1	1.00
205			300	124	301	123	1	1	1.41

206			297	124	297	123	0	1	1.00
207			293	124	293	123	0	1	1.00
208			288	124	288	123	0	1	1.00
209			283	123	282	122	1	1	1.41
210			277	123	277	122	0	1	1.00
211			269	123	269	123	0	0	0.00
212			260	123	259	122	1	1	1.41
213			250	123	250	121	0	2	2.00
214	26	Out	260	83	260	82	0	1	1.00
215			271	78	272	77	1	1	1.41
216			281	73	280	71	1	2	2.24
217			289	70	289	68	0	2	2.00
218			294	67	294	68	0	1	1.00
219			300	64	299	63	1	1	1.41
220			304	62	302	61	2	1	2.24
221			307	60	306	60	1	0	1.00
222			310	58	310	59	0	1	1.00
223			313	56	313	56	0	0	0.00
224	27	In	317	109	316	108	1	1	1.41
225			308	110	308	109	0	1	1.00
226			295	110	294	109	1	1	1.41
227			281	112	281	112	0	0	0.00
228			263	113	263	113	0	0	0.00
229	28	Out	216	62	216	60	0	2	2.00
230			217	50	217	49	0	1	1.00
231			219	37	218	35	1	2	2.24
232			218	25	217	24	1	1	1.41
233			220	14	220	14	0	0	0.00
234	29	In	317	112	317	111	0	1	1.00
235			309	112	309	111	0	1	1.00
236			296	113	296	113	0	0	0.00
237			282	114	281	113	1	1	1.41
238			264	115	264	114	0	1	1.00
239	30	Out	253	145	253	144	0	1	1.00
240			266	153	265	152	1	1	1.41
241			272	157	273	155	1	2	2.24
242			277	160	276	159	1	1	1.41
243			281	163	280	162	1	1	1.41
244			285	166	283	166	2	0	2.00
245			288	169	287	167	1	2	2.24
246			290	172	289	171	1	1	1.41
247			292	174	292	174	0	0	0.00
248			293	176	293	175	0	1	1.00
249			294	179	293	178	1	1	1.41
250			295	181	294	180	1	1	1.41
251			295	182	296	180	1	2	2.24
252			295	184	294	182	1	2	2.24
253			295	184	294	182	1	2	2.24
254	31	In	316	111	316	110	0	1	1.00
255			306	110	306	110	0	0	0.00
256			293	112	293	111	0	1	1.00
257			278	112	278	111	0	1	1.00
258			259	113	258	112	1	1	1.41

259	32	Out	122	97	121	96	1	1	1.41
260			108	94	108	94	0	0	0.00
261			96	92	94	90	2	2	2.83
262			82	89	82	88	0	1	1.00
263			68	86	68	85	0	1	1.00
264			54	84	54	83	0	1	1.00
265			42	82	41	82	1	0	1.00
266			30	81	30	80	0	1	1.00
267			19	80	18	79	1	1	1.41
268			10	79	9	79	1	0	1.00
269			3	79	2	78	1	1	1.41
270	33	In	317	118	316	117	1	1	1.41
271			308	119	307	118	1	1	1.41
272			297	121	297	120	0	1	1.00
273			284	123	283	122	1	1	1.41
274			268	126	267	125	1	1	1.41
275			260	124	260	125	0	1	1.00
276	34	Out	206	68	205	66	1	2	2.24
277			203	65	202	64	1	1	1.41
278			200	63	199	62	1	1	1.41
279			197	61	196	59	1	2	2.24
280			200	58	199	56	1	2	2.24
281			201	54	201	53	0	1	1.00
282			202	50	202	48	0	2	2.00
283			203	47	202	46	1	1	1.41
284			203	43	203	43	0	0	0.00
285			203	40	202	40	1	0	1.00
286			203	38	202	36	1	2	2.24
287			202	35	201	34	1	1	1.41
288			201	33	200	31	1	2	2.24
289			199	31	198	29	1	2	2.24
290			197	29	197	28	0	1	1.00
291			196	27	195	27	1	0	1.00
292			194	26	194	26	0	0	0.00
293			192	25	189	24	3	1	3.16
294			191	22	191	22	0	0	0.00
295			192	20	191	19	1	1	1.41
296			192	17	191	16	1	1	1.41
297			191	14	191	13	0	1	1.00
298			190	12	189	12	1	0	1.00
299			190	10	189	9	1	1	1.41
300			188	8	187	8	1	0	1.00
301			187	6	187	5	0	1	1.00
302			185	6	185	5	0	1	1.00
303			185	3	185	2	0	1	1.00
304	35	In	316	111	316	110	0	1	1.00
305			309	112	308	111	1	1	1.41
306			296	112	296	111	0	1	1.00
307			283	113	283	112	0	1	1.00
308			266	114	266	114	0	0	0.00
309			244	113	244	112	0	1	1.00
310	36	Out	293	134	292	134	1	0	1.00
311			305	137	305	136	0	1	1.00

312			310	138	310	137	0	1	1.00
313	37	In	316	115	316	114	0	1	1.00
314			304	116	304	116	0	0	0.00
315			288	117	288	116	0	1	1.00
316			270	119	270	118	0	1	1.00
317			248	120	247	119	1	1	1.41
318	38	Out	259	126	258	125	1	1	1.41
319			276	126	275	125	1	1	1.41
320			284	126	283	126	1	0	1.00
321	39	In	316	104	315	102	1	2	2.24
322			304	103	304	101	0	2	2.00
323			290	103	290	102	0	1	1.00
324			274	102	272	100	2	2	2.83
325			255	102	256	102	1	0	1.00
326	40	Out	165	54	164	53	1	1	1.41
327			148	45	148	47	0	2	2.00
328	41	In	311	117	311	116	0	1	1.00
329			298	118	297	116	1	2	2.24
330			283	120	283	118	0	2	2.00
331			264	122	263	121	1	1	1.41
332	42	Out	255	118	255	117	0	1	1.00
333			290	110	290	109	0	1	1.00
334			306	107	306	105	0	2	2.00
335	43	In	315	115	314	114	1	1	1.41
336			306	116	306	115	0	1	1.00
337			294	117	293	116	1	1	1.41
338			280	118	281	117	1	1	1.41
339			264	120	264	118	0	2	2.00
340	44	Out	265	137	265	135	0	2	2.00
341			276	138	275	137	1	1	1.41
342			284	140	284	139	0	1	1.00
343			291	140	291	139	0	1	1.00
344			297	142	295	141	2	1	2.24
345			302	142	302	142	0	0	0.00
346			306	142	306	141	0	1	1.00
347			309	143	308	142	1	1	1.41
348			312	143	312	142	0	1	1.00
349	45	In	316	118	316	118	0	0	0.00
350			306	119	306	118	0	1	1.00
351			294	122	293	121	1	1	1.41
352			280	124	280	123	0	1	1.00
353			264	127	264	126	0	1	1.00
354	46	Out	166	63	165	61	1	2	2.24
355			153	57	151	56	2	1	2.24
356			138	52	137	50	1	2	2.24
357			123	46	123	45	0	1	1.00
358			106	41	105	41	1	0	1.00
359			90	37	89	36	1	1	1.41
360			74	34	74	33	0	1	1.00
361			59	31	59	29	0	2	2.00
362			45	30	43	29	2	1	2.24
363			32	30	32	29	0	1	1.00
364			21	30	21	29	0	1	1.00

365			11	31	9	31	2	0	2.00
366			4	32	4	32	0	0	0.00
367	47	IN	253	132	252	131	1	1	1.41
368			245	134	245	132	0	2	2.00
369			234	136	235	136	1	0	1.00
370			220	139	220	138	0	1	1.00
371	48	OUT	157	206	155	208	2	2	2.83
372			157	231	157	230	0	1	1.00
373			163	238	164	236	1	2	2.24
374	49	IN	304	110	303	109	1	1	1.41
375			294	109	292	108	2	1	2.24
376			281	108	279	107	2	1	2.24
377			266	107	267	106	1	1	1.41
378			247	107	246	105	1	2	2.24
379			222	106	223	105	1	1	1.41
380	50	OUT	114	31	114	32	0	1	1.00
381			105	12	105	12	0	0	0.00
382	51	IN	299	122	299	121	0	1	1.00
383			288	122	287	121	1	1	1.41
384			273	120	274	119	1	1	1.41
385			255	119	255	118	0	1	1.00
386			233	118	233	116	0	2	2.00
387	52	OUT	184	67	183	66	1	1	1.41
388			202	53	201	53	1	0	1.00
389			216	42	217	41	1	1	1.41
390			226	33	226	32	0	1	1.00
391	53	IN	317	124	317	124	0	0	0.00
392			302	124	301	123	1	1	1.41
393			293	125	292	123	1	2	2.24
394			282	124	281	123	1	1	1.41
395			266	124	265	123	1	1	1.41
396			248	126	249	124	1	2	2.24
397			233	131	233	130	0	1	1.00
398	54	OUT	46	70	46	68	0	2	2.00
399			14	60	13	59	1	1	1.41
400	55	IN	314	118	313	117	1	1	1.41
401			306	118	306	118	0	0	0.00
402			298	118	297	116	1	2	2.24
403			286	118	286	117	0	1	1.00
404			272	119	272	118	0	1	1.00
405			246	131	246	130	0	1	1.00
406			210	121	210	120	0	1	1.00
407	56	OUT	211	88	211	87	0	1	1.00
408			237	79	237	79	0	0	0.00
409			250	74	236	78	14	4	14.56
410			266	69	251	73	15	4	15.52
411			274	66	266	70	8	4	8.94
412			279	65	279	64	0	1	1.00
413			284	63	284	63	0	0	0.00
414			288	61	284	63	4	2	4.47
415			291	60	288	61	3	1	3.16
416			294	58	290	59	4	1	4.12
417			297	57	294	58	3	1	3.16

418			299	56	297	57	2	1	2.24
419			300	56	300	55	0	1	1.00
420	57	IN	315	120	315	119	0	1	1.00
421			308	120	308	119	0	1	1.00
422			298	120	297	120	1	0	1.00
423			286	121	285	120	1	1	1.41
424			272	122	272	121	0	1	1.00
425			253	123	253	122	0	1	1.00
426			230	124	229	123	1	1	1.41
427	58	OUT	217	114	218	113	1	1	1.41
428			237	112	236	112	1	0	1.00
429			253	110	253	109	0	1	1.00
430			266	108	264	107	2	1	2.24
431			276	106	276	105	0	1	1.00
432			285	106	286	105	1	1	1.41
433			294	104	296	103	2	1	2.24
434			298	96	298	96	0	0	0.00
435	59	IN	312	129	311	128	1	1	1.41
436			304	130	304	130	0	0	0.00
437			293	130	293	129	0	1	1.00
438			280	131	279	129	1	2	2.24
439			254	144	253	143	1	1	1.41
440			223	142	224	141	1	1	1.41
441	60	OUT	202	89	203	86	1	3	3.16
442			232	79	233	78	1	1	1.41
443			246	74	246	73	0	1	1.00
444			274	64	274	64	0	0	0.00
445			290	59	291	57	1	2	2.24
446			300	55	299	54	1	1	1.41
447			304	54	304	54	0	0	0.00
448			307	52	308	51	1	1	1.41
449			310	52	311	51	1	1	1.41
450	61	IN	317	113	317	112	0	1	1.00
451			303	112	303	111	0	1	1.00
452			294	111	293	109	1	2	2.24
453			282	110	281	109	1	1	1.41
454			267	109	267	107	0	2	2.00
455			246	114	247	113	1	1	1.41
456			232	111	232	109	0	2	2.00
457	62	OUT	44	62	45	61	1	1	1.41
458			10	58	11	57	1	1	1.41
459	63	IN	297	132	299	132	2	0	2.00
460			294	133	295	132	1	1	1.41
461			288	136	288	134	0	2	2.00
462			286	137	284	136	2	1	2.24
463			273	142	271	142	2	0	2.00
464			264	146	261	144	3	2	3.61
465			250	150	250	148	0	2	2.00
466			235	154	235	152	0	2	2.00
467			214	160	213	158	1	2	2.24
468	64	OUT	43	90	43	89	0	1	1.00
469			19	92	20	93	1	1	1.41
470			14	78	14	80	0	2	2.00

471	65	IN	316	124	317	123	1	1	1.41
472			308	124	309	123	1	1	1.41
473			300	126	300	127	0	1	1.00
474			289	128	289	128	0	0	0.00
475			275	131	274	130	1	1	1.41
476			266	133	267	132	1	1	1.41
477			263	131	262	130	1	1	1.41
478			256	132	256	131	0	1	1.00
479			245	133	245	132	0	1	1.00
480			236	134	237	133	1	1	1.41
481			220	135	219	134	1	1	1.41
482	66	OUT	205	170	205	170	0	0	0.00
483			229	184	230	182	1	2	2.24
484			237	190	238	189	1	1	1.41
485			252	202	252	201	0	1	1.00
486			256	204	256	203	0	1	1.00
487			266	208	265	208	1	0	1.00
488			272	210	271	208	1	2	2.24
489			276	212	276	212	0	0	0.00
490	67	IN	315	122	315	121	0	1	1.00
491			308	123	306	123	2	0	2.00
492			299	124	299	123	0	1	1.00
493			288	126	289	124	1	2	2.24
494			274	128	274	127	0	1	1.00
495			266	134	265	133	1	1	1.41
496			264	128	265	127	1	1	1.41
497			248	129	249	128	1	1	1.41
498			236	129	235	128	1	1	1.41
499			220	130	219	129	1	1	1.41
500	68	OUT	214	101	215	98	1	3	3.16
501			236	94	237	93	1	1	1.41
502			245	92	246	91	1	1	1.41
503			258	89	258	88	0	1	1.00
504			264	88	264	87	0	1	1.00
505			268	86	267	85	1	1	1.41
506			272	85	271	85	1	0	1.00
507			276	84	276	84	0	0	0.00
508			278	84	278	83	0	1	1.00
509	69	IN	255	131	254	130	1	1	1.41
510			250	130	250	128	0	2	2.00
511			245	130	244	130	1	0	1.00
512			239	130	239	130	0	0	0.00
513			232	129	231	128	1	1	1.41
514			223	129	222	128	1	1	1.41
515	70	OUT	213	104	213	104	0	0	0.00
516			223	102	224	102	1	0	1.00
517			228	101	230	100	2	1	2.24
518			232	101	234	101	2	0	2.00
519			236	100	237	99	1	1	1.41
520			239	99	240	98	1	1	1.41
521			242	99	243	99	1	0	1.00
522			244	98	245	98	1	0	1.00
523			248	98	249	97	1	1	1.41

524			250	98	251	98	1	0	1.00
525	71	IN	307	130	308	128	1	2	2.24
526			304	130	304	129	0	1	1.00
527			300	130	299	130	1	0	1.00
528			296	130	296	129	0	1	1.00
529			292	131	292	129	0	2	2.00
530			285	131	284	130	1	1	1.41
531			277	131	276	130	1	1	1.41
532			267	132	268	132	1	0	1.00
533			254	133	255	132	1	1	1.41
534			238	133	240	132	2	1	2.24
535			218	134	219	133	1	1	1.41
536	72	OUT	120	199	120	197	0	2	2.00
537			123	218	122	217	1	1	1.41
538			134	233	135	233	1	0	1.00
539	73	IN	270	132	269	132	1	0	1.00
540			264	133	262	132	2	1	2.24
541			256	135	254	134	2	1	2.24
542			246	137	244	135	2	2	2.83
543			232	139	231	138	1	1	1.41
544			213	143	214	142	1	1	1.41
545	74	OUT	64	79	62	78	2	1	2.24
546			47	74	45	71	2	3	3.61
547	75	IN	315	125	317	124	2	1	2.24
548			300	126	301	125	1	1	1.41
549			292	128	292	128	0	0	0.00
550			286	128	285	128	1	0	1.00
551			282	129	282	128	0	1	1.00
552			278	129	277	128	1	1	1.41
553			273	130	273	128	0	2	2.00
554			265	131	264	130	1	1	1.41
555			256	132	257	130	1	2	2.24
556			243	133	244	132	1	1	1.41
557			226	134	228	133	2	1	2.24
558	76	OUT	188	188	189	186	1	2	2.24
559			199	214	197	213	2	1	2.24
560			203	224	203	224	0	0	0.00
561	77	IN	306	136	306	135	0	1	1.00
562			297	138	297	138	0	0	0.00
563			285	142	286	141	1	1	1.41
564			278	145	278	144	0	1	1.00
565			277	143	276	142	1	1	1.41
566			266	147	266	145	0	2	2.00
567			258	150	257	150	1	0	1.00
568			249	152	248	153	1	1	1.41
569			236	155	235	155	1	0	1.00
570			220	159	219	158	1	1	1.41
571	78	OUT	76	146	75	145	1	1	1.41
572			36	140	35	139	1	1	1.41
573	79	IN	288	137	288	137	0	0	0.00
574			284	139	284	139	0	0	0.00
575			280	141	279	141	1	0	1.00
576			274	142	273	142	1	0	1.00

577			267	145	267	144	0	1	1.00
578			258	148	257	148	1	0	1.00
579			246	151	246	150	0	1	1.00
580			230	155	230	155	0	0	0.00
581			210	159	210	157	0	2	2.00
582	80	OUT	177	198	178	198	1	0	1.00
583			193	234	194	232	1	2	2.24
584	81	IN	317	121	317	120	0	1	1.00
585			311	122	311	121	0	1	1.00
586			302	123	303	123	1	0	1.00
587			292	124	292	123	0	1	1.00
588			278	125	278	123	0	2	2.00
589			269	132	268	131	1	1	1.41
590			269	124	269	124	0	0	0.00
591			254	124	254	124	0	0	0.00
592			242	123	242	122	0	1	1.00
593			227	122	228	122	1	0	1.00
594	82	OUT	74	81	73	80	1	1	1.41
595			46	56	45	52	1	4	4.12
596			23	35	23	34	0	1	1.00
597	83	IN	317	122	317	121	0	1	1.00
598			312	124	311	123	1	1	1.41
599			304	125	306	124	2	1	2.24
600			294	128	294	128	0	0	0.00
601			289	130	288	130	1	0	1.00
602			282	140	281	138	1	2	2.24
603			284	129	283	128	1	1	1.41
604			267	132	266	131	1	1	1.41
605			258	133	256	132	2	1	2.24
606			246	134	246	132	0	2	2.00
607			230	136	231	135	1	1	1.41
608	84	OUT	64	131	64	128	0	3	3.00
609			46	132	46	130	0	2	2.00
610	85	IN	316	125	315	123	1	2	2.24
611			309	127	306	127	3	0	3.00
612			300	128	299	128	1	0	1.00
613			290	130	290	130	0	0	0.00
614			277	132	277	132	0	0	0.00
615			262	136	262	136	0	0	0.00
616			241	139	240	137	1	2	2.24
617			225	145	224	144	1	1	1.41
618	86	OUT	227	209	228	208	1	1	1.41
619			239	220	239	219	0	1	1.00
620			248	229	249	228	1	1	1.41

Table A-3: Experiment results of accuracy test

A.1.3 Efficiency Experiment Results

Test Video Footage 1		Test Video Footage 2	
AMD 1.54GHz	INTEL 3.2GHz	AMD 1.54GHz	INTEL 3.2GHz
33.46	59.00	31.00	64.00
32.00	64.00	32.48	64.00
32.48	62.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
27.00	44.00	19.00	30.00
32.00	64.00	32.00	64.00
32.00	61.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	63.00
32.00	64.00	32.00	63.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	62.00	32.00	64.00
32.00	62.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	61.00
32.00	64.00	30.51	64.00
32.00	64.00	32.51	64.00
32.00	64.00	32.00	64.00
32.00	40.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	51.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	63.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00

32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
33.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	63.00	32.00	64.00
32.00	64.00	32.00	64.00
33.00	64.00	33.00	64.00
32.00	64.00	35.00	64.00
32.00	64.00	32.00	64.00
32.00	62.01	32.00	64.00
32.00	62.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	62.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00

32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	64.00
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00	62.00	32.00	
32.00	62.00	32.00	
32.00	61.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.48	
32.00	64.00	32.00	
32.00	64.00	32.48	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.51	64.00	32.00	
32.00	64.00	32.00	
32.48	64.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00	64.00	32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	

32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.51	
32.00		32.00	
32.00		32.48	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
33.00		32.00	
33.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	

32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		32.00	
33.00		32.00	
32.00		32.00	
32.00		32.00	
32.00		24.94	
32.00		43.00	
32.00		43.00	
32.00		45.00	
32.00		44.29	
32.00		44.00	
32.00		35.47	
32.00		45.00	
32.00		44.00	
32.00		44.00	
32.00		43.31	
32.00		44.00	
32.00		44.00	
32.00			
32.00			
32.00			
33.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			

32.00			
33.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			
32.00			

Table A-4: The sampled frame rate in efficiency experiment.